NAVAL POSTGRADUATE SCHOOL

MONTEREY, CALIFORNIA

THESIS

COMMUNICATONS NETWORK DESIGN, SIMULATION, AND ANALYSIS FOR AN AUTONOMOUS UNMANNED VEHICLE SYSTEM

by

Nathan C. Matson

June 2011

Thesis Co-Advisors: Clifford Whitcomb
Weilian Su

Approved for public release; distribution is unlimited

THIS PAGE INTENTIONALLY LEFT BLANK

REPORT DOCUMENTATION PAGE			Form Approved OMB No. 0704-0188
Public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instruction, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden, to Washington headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302, and to the Office of Management and Budget, Paperwork Reduction Project (0704-0188) Washington DC 20503.			
1. AGENCY USE ONLY *(Leave blank)*	**2. REPORT DATE** June 2011	**3. REPORT TYPE AND DATES COVERED** Master's Thesis	
4. TITLE AND SUBTITLE Communications Network Design, Simulation, and Analysis for an Autonomous Unmanned Vehicle System		**5. FUNDING NUMBERS**	
6. AUTHOR(S) Nathan C. Matson			
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) Naval Postgraduate School Monterey, CA 93943-5000		**8. PERFORMING ORGANIZATION REPORT NUMBER**	
9. SPONSORING /MONITORING AGENCY NAME(S) AND ADDRESS(ES) N/A		**10. SPONSORING/MONITORING AGENCY REPORT NUMBER**	
11. SUPPLEMENTARY NOTES The views expressed in this thesis are those of the author and do not reflect the official policy or position of the Department of Defense or the U.S. Government. IRB Protocol number: N/A			
12a. DISTRIBUTION / AVAILABILITY STATEMENT Approved for public release; distribution is unlimited.		**12b. DISTRIBUTION CODE** A	

13. ABSTRACT (maximum 200 words)

In this thesis, we designed, simulated, and analyzed a wireless communications system for an autonomous unmanned vehicle system. The system used for the design context is the Unmanned Vehicle (UV) Sentry, which is a system of autonomous unmanned vehicles that can be tasked for a variety of missions that involve the patrolling and protecting of a geographical region. Accordingly, the communications network needs to allow for flexibility of the vehicle topography to enable large amounts of delay intolerant sensor data to be transmitted between nodes and a capability that allows vehicles to act as relays for other vehicles. To meet these requirements, a medium access control (MAC) relay protocol based on the IEEE 802.16 standard was developed. To evaluate the performance of the protocol, Simulink was used to model the performance of the protocol as it was implemented in a specific UV Sentry scenario. Several network parameters were chosen as factors for the model, and these factors were systematically varied to yield a full factorial design of experiments. The network quality of service parameters for the tests were then analyzed to determine the best communication network configuration for the UV Sentry scenario and to illuminate the tradeoffs between the factors.

14. SUBJECT TERMS unmanned vehicles, UV Sentry, WiMAX, 802.16, 802.16j, relay, multihop, Simulink, medium access control, quality of service, design of experiments			**15. NUMBER OF PAGES** 241
			16. PRICE CODE
17. SECURITY CLASSIFICATION OF REPORT Unclassified	**18. SECURITY CLASSIFICATION OF THIS PAGE** Unclassified	**19. SECURITY CLASSIFICATION OF ABSTRACT** Unclassified	**20. LIMITATION OF ABSTRACT** UU

NSN 7540-01-280-5500

Standard Form 298 (Rev. 2-89)
Prescribed by ANSI Std. 239-18

THIS PAGE INTENTIONALLY LEFT BLANK

Approved for public release; distribution is unlimited

COMMUNICATONS NETWORK DESIGN, SIMULATION, AND ANALYSIS FOR AN AUTONOMOUS UNMANNED VEHICLE SYSTEM

Nathan C. Matson
Lieutenant, United States Navy
B.S., United States Naval Academy, 2002

Submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE IN SYSTEMS ENGINEERING
AND
MASTER OF SCIENCE IN ELECTRICAL ENGINEERING

from the

NAVAL POSTGRADUATE SCHOOL
June 2011

Author: Nathan C. Matson

Approved by: Clifford Whitcomb
Thesis Co-Advisor

Weilian Su
Thesis Co-Advisor

Clifford Whitcomb
Chairman, Department of Systems Engineering

R. Clark Robertson
Chairman, Department of Electrical and Computer Engineering

THIS PAGE INTENTIONALLY LEFT BLANK

ABSTRACT

In this thesis, we designed, simulated, and analyzed a wireless communications system for an autonomous unmanned vehicle system. The system used for the design context is the Unmanned Vehicle (UV) Sentry, which is a system of autonomous unmanned vehicles that can be tasked for a variety of missions that involve the patrolling and protecting of a geographical region. Accordingly, the communications network needs to allow for flexibility of the vehicle topography to enable large amounts of delay intolerant sensor data to be transmitted between nodes and a capability that allows vehicles to act as relays for other vehicles. To meet these requirements, a medium access control (MAC) relay protocol based on the IEEE 802.16 standard was developed. To evaluate the performance of the protocol, Simulink was used to model the performance of the protocol as it was implemented in a specific UV Sentry scenario. Several network parameters were chosen as factors for the model, and these factors were systematically varied to yield a full factorial design of experiments. The network quality of service parameters for the tests were then analyzed to determine the best communication network configuration for the UV Sentry scenario and to illuminate the tradeoffs between the factors.

THIS PAGE INTENTIONALLY LEFT BLANK

TABLE OF CONTENTS

THIS PAGE INTENTIONALLY LEFT BLANK

LIST OF FIGURES

THIS PAGE INTENTIONALLY LEFT BLANK

LIST OF TABLES

THIS PAGE INTENTIONALLY LEFT BLANK

LIST OF ACRONYMS AND ABBREVIATIONS

3G	Third Generation
4G	Fourth Generation
AFRICOM	United States African Command
ARQ	Automatic Repeat Request
BE	Best-Effort Service
BPSK	Binary Phase-Shift Keying
BS	Base Station
BW	Bandwidth
CBRN	Chemical, Biological, Radiological, or Nuclear
CDMA	Code-Division Multiple Access
CENTCOM	United States Central Command
CID	Connection Identifier
COI	Contact of Interest
COP	Common Operational Picture
CSMA	Carrier Sense Multiple Access
CTS	Clear-to-Send
D	Distance Factor
DL	Downlink
EO	Electro-Optical
ErtPS	Extended Real-Rime Polling Service
EV-DO	Evolution-Data Optimized
FCH	Frame Control Header
FDMA	Frequency-Division Multiple Access
FLIR	Forward-Looking Infrared
HSPA	High Speed Packet Access
IEEE	Institute of Electrical and Electronics Engineers
I	Interference Factor

IR	Infrared
LCS	Littoral Combat Ship
LOS	Line-of-Sight
LTE	Long Term Evolution
MAP	Map
MPDU	Medium Access Control Protocol Data Units
MSDU	Medium Access Control Service Data Units
NLOS	NonLine-of-Sight
NRT	Near Real Time
nrtPS	NonReal-Time Polling Service
OFDM	Orthogonal Frequency-Division Multiplexing
OFDMA	Orthogonal Frequency-Division Multiple Access
PDU	Protocol Data Unit
QAM	Quadrature Amplitude Modulation
QoS	Quality of Service
QPSK	Quadrature Phase-Shift Keying
R	Relay Factor
RF	Radio Frequency
ROE	Rules of Engagement
RS	Relay Station
rtPS	Real-Time Polling Service
RTS	Request-to-Send
S	Bandwidth Factor
SDMA	Space-Division Multiple Access
SFID	Service Flow Identifier
SIGINT	Signals Intelligence
SNIR	Signal-to-Noise-Plus-Interference Ratio
SNR	Signal-to-Noise Ratio
SS	Subscriber Station

T	FLIRs Factor
TDMA	Time-Division Multiple Access
UAV	Unmanned Aerial Vehicle
UGS	Unsolicited Grant Services
UL	Uplink
UMTS	Universal Mobile Telephone System
UNTL	Universal Naval Task List
USV	Unmanned Surface Vehicle
USW	Undersea Warfare
UUV	Unmanned Underwater Vehicle
UV	Unmanned Vehicle
VTUAV	Vertical Takeoff Unmanned Aerial Vehicle
WiMAX	Worldwide Interoperability for Microwave Access

THIS PAGE INTENTIONALLY LEFT BLANK

EXECUTIVE SUMMARY

Unmanned vehicles are increasingly replacing humans and human operated vehicles on the battlefield and in surveillance roles. In the current situation, the unmanned vehicles are heavily reliant on their human operators to manually control the vehicle. However, the UV (Unmanned Vehicle) Sentry system proposes to remove the need for direct human control of unmanned vehicles. The UV Sentry system is composed of multiple autonomous, unmanned vehicles that can provide surveillance, detection and engagement capability to a large maritime region [1]. The vehicles will collectively coordinate their own decisions and then act upon them. The human will occupy a supervisory role on a central operational platform and will be able to monitor the UV Sentry systems actions.

The communications requirements for such a system may vary widely. If the UV Sentry is operating in fully autonomous mode, the communications network requirements will likely be minimal as the vehicles will only transmit and receive coordination messages. In this mode the UV Sentry operator will be able to view the current tasking of the vehicles and a summary of the surveillance sensor data. However, if the operator desires more detailed information on a contact, he will be able to request to view the raw sensor data [1]. He will also have the ability to override the vehicle autonomy and assume manual control of the UV Sentry system. In this mode, the network is likely to be strained as the near real time (NRT) sensor messages will have strict quality of service (QoS) requirements. While the communications network may spend much of its time underutilized, it should be capable of handling potentially high data requirements with little allowable delay.

Because UV Sentry can be deployed in many different scenarios, a specific scenario was chosen for the communications network design and modeling. The scenario chosen was the protection of a maritime oil platform against a high-speed small boat attack. In this scenario, two airborne and four surface UV Sentry vehicles patrolled a 7 km circle. By using the UV Sentry operational activities, we were able to identify the

specific communications requirements for UV Sentry in this scenario, and through the use of the UNTL metrics, we highlighted the metrics important to the UV Sentry operator.

After surveying the possible network architectures and standards in order to find the network type that best matched the UV Sentry needs, we chose the IEEE 802.16 standard. However, there are many variations of the 802.16 standard, so we examined these variations in detail at the physical and MAC layers. In particular, the use of relays that could extend the network range was important for UV Sentry. However, we found that the current implementations of the IEEE 802.16j relay standard were not optimal for use in UV Sentry. Accordingly, we suggested an alternate relay implementation of the 802.16j standard. The alternate relay implementation gives a high level of autonomy to the relay vehicles so that in many ways the relay vehicles act as a central base station (BS) for an 802.16 network. This permits the UV Sentry network to extend its range without allowing the relay stations to cause interference. However, this suggested relay implementation needed to be evaluated.

In order to evaluate the chosen network's ability to facilitate UV Sentry communications, a model of both a relay and nonrelay network was built in Simulink. In order to test the network's performance, we chose the five key factors of vehicle distance, interference level, offered load, available bandwidth, and relay capability. We then conducted a full factorial design of experiments on the network by systematically changing these variable values between their minimum and maximum. Collected data on the uplink and downlink connections was first analyzed by gauging the networks response as it was measured by the key metrics of delay and dropped packets. Furthermore, the overall network error rates, throughput, and ratio of goodput to throughput were evaluated to ensure that these valuables were reasonable. In so doing, we identified the most important factors for the UV Sentry Network and then recommended a UV Sentry network configuration based on the factorial tests.

The ultimate recommended UV Sentry communications network was a hybrid of two possible network configurations. Since the first two factors of distance and interference affected the signal-to-noise-plus-interference ratio (SNIR) and, thus, were

independent environmental factors, they could not be included as configuration possibilities. The resultant network recommendations were based on the combination of the offered load, available bandwidth, and relay capability. For operations where direct line-of-sight (LOS) communications between the transmitting vehicle and the central operating platform is possible, this mode should be used. In this case, the network can support two separate forward-looking infrared (FLIR) video transmissions at 7 km with interference while on a reduced bandwidth of 5 MHz. However, if LOS communications between the central base station (BS) and the vehicle are not possible, then the relay mode should be utilized. In relay mode, two FLIR transmissions at 7 km with interference require the full 10 MHz bandwidth. Accordingly, most of the time, the network can operate on a reduced spectrum; however, if multiple FLIR transmissions are required with the use of a relay, then the 10 MHz bandwidth should be available.

To demonstrate the relay network performance, an intercept scenario was conducted where two surface UV Sentry vehicles intercept a surface threat at 7 km away from the oil platform that was co-located with the BS. The UV Sentry vehicles pursue the threat from 7 km to 2 km and transmit their network traffic through an airborne UV Sentry vehicle relay. The resulting performance gave empirical evidence that the network can handle the offered load of two FLIR videos with minimal delay and dropped packets and that the network performance increases as the vehicles approach the BS. Accordingly, the recommended network seems to be capable of meeting the UV Sentry needs for the given scenario.

LIST OF REFERENCES

[1] Office of Naval Research, "UV sentry system innovative naval prototype proposal overview," Office of Naval Research, Arlington, VA, Dec. 2008.

THIS PAGE INTENTIONALLY LEFT BLANK

ACKNOWLEDGMENTS

I would like to express my sincere gratitude and appreciation to my co-advisors Dr. Weilian Su and Dr. Clifford Whitcomb. Their patience, guidance, and insight have been invaluable during the thesis writing process. Additionally, without the quality instruction I received in their classes, this thesis would not have been possible. I hope the reader find this thesis to be a reflection of the consistently high standards they set at the Naval Postgraduate School.

I would also like to thank my family for their unwavering support throughout the thesis process.

THIS PAGE INTENTIONALLY LEFT BLANK

I. INTRODUCTION

The Unmanned Vehicle (UV) Sentry and the need for a communications network that uniquely meets its communications requirements are introduced in this chapter. Related research on the development of communications networks for similar applications is then discussed. Lastly, the organization of the thesis is discussed.

A. BACKGROUND

Unmanned vehicles are increasingly replacing humans and human operated vehicles on the battlefield and in surveillance roles. There are several advantages to removing the human from these situations. Human life support systems are unnecessary on unmanned vehicles, which allows the vehicles to remain on scene longer and with more capability. Furthermore, the loss of a vehicle is more acceptable if a human is not onboard [1]. While the advent of the unmanned vehicle has been able to remove the human from the scene, the human is still critical to mission success. For example, a Predator unmanned aerial vehicle (UAV) can conduct surveillance and engage targets without the presence of a human in the aircraft. However, there is a team of humans at a remote operating station that are busy remotely piloting the aircraft, watching the surveillance footage, and making mission-critical decisions concerning the UAV's next tasking [1]. In the current situation, the unmanned vehicles are heavily reliant on their human operators to manually control the vehicle.

In contrast, the UV Sentry system proposes to remove the need for direct human control of unmanned vehicles. The UV Sentry system is composed of multiple autonomous, unmanned vehicles that can provide surveillance, detection and engagement capability to a large maritime region [2]. The vehicles will collectively coordinate their own decisions and act upon them. The human will occupy a supervisory role on a central operational platform and will be able to monitor the UV Sentry systems actions.

The communications requirements for such a system will vary widely. If the UV Sentry is operating in fully autonomous mode, the communications network requirements

will likely be minimal as the vehicles will only transmit and receive coordination messages. In this mode the UV Sentry operator will be able to view the current tasking of the vehicles and a summary of the surveillance sensor data. However, if the operator desires more detailed information on a contact, he will be able to request to view the raw sensor data [2]. He will have the ability to override the vehicle autonomy and assume manual control of the UV Sentry system. In this mode, the network requirements are likely to be severely taxed as the near real time (NRT) sensor messages will have strict quality of service (QoS) requirements. While the communications network may spend much of its time underutilized, it should be capable of handling potentially high requirements.

B. RELATED WORK

The IEEE (Institute of Electrical and Electronics Engineers) 802.16 Standard seems to be the most preferred network standard for the purpose of deploying a high data rate communications networks that is capable of serving a large area. In particular, two separate implementations of the 802.16 standard allow it to expand the network coverage. The 802.16d mesh implementation allows multiple numbers of fixed network nodes to communicate among each other without requiring the control of a central base station (BS). In contrast, the 802.16j amendment to the 802.16e standard allows a central BS to utilize relay stations to expand its network coverage through the use of relay stations.

References [3], [4], and [5] contain descriptions of the implementation of the 802.16d mesh protocol in simulated scenarios to allow the vehicles to communicate in a decentralized, contention-based manner. The resulting network architecture is robust because it allows multiple pathways between nodes so if a single network node is lost the network can use pre-existing pathways around that node [3]. However, routing for mobile mesh nodes can introduce large amounts of overhead and congestion – leading to QoS challenges [3]. The use of mesh protocol to allow merchant vessels to create a mesh network that would allow distant offshore ships to connect to shore-based back-haul communications was simulated in [4]. However, for fast moving vehicles, it was shown

in [5] that the 802.16d mesh protocol had significant throughput and latency issues because the protocol was not designed to be used by mobile users.

Use of 802.16j relays to facilitate the communication between vehicles in a tactical network is discussed in [6] and [7]. In contrast to the mesh approach, these vehicles communicate according to a schedule promulgated by a central BS. The BS used the relays to either extend the range of the network or to increase the signal to noise ratio (SNR) for the mobile vehicles. Furthermore, these networks proved capable of guaranteeing network QoS and increased network performance for extended range deployments of mobile users [7].

The use of a hybrid of the 802.16d and 802.16e networks is discussed in [8]. In this network, fixed mesh 802.16d stations provide a backhaul capability that can be deployed to set up a stationary tactical network. These mesh stations then provide connection to mobile vehicles in the vicinity through a separate 802.16e radio that is interconnected with the 802.16d radio.

A unique solution to provide high data rate capability to U. S. Navy ships by using directional antennas to create spatially aware ad-hoc networks is proposed in [9]. These directional antennas provide an increased gain in the direction that they are pointed. This allows for higher data rates on both 802.11 and 802.16 networks. Nevertheless, these directional antennas introduce additional complexities at both the medium access control (MAC) and physical layers.

C. OBJECTIVE AND APPROACH

The objective of this thesis is to first choose a network type that is best suited to fulfill the requirements of UV Sentry and then demonstrate the networks suitability through the use of a model. Initially, multiple different network standards were compared against each other, and based on the demonstrated performance of these standards, a network standard was chosen for UV Sentry. Specifically, the 802.16 standard was chosen based on its ability to cover a large area with a high data rate and

because of the standards flexibility in implementation [10]. Furthermore, a novel relay implementation of the standard was developed to enhance the standard's ability to provide coverage at long ranges.

In order to be assured that the 802.16 standard can provide acceptable QoS to the UV Sentry system in different situations, a full factorial design of experiments was conducted. In this design of experiments, the five network factors of distance, interference, number of forward-looking infrared (FLIR) videos, available bandwidth (BW), and relay capability were varied between their high and low values. The network response to these variations was evaluated, and ultimately, a network configuration for UV Sentry was recommended.

D. THESIS ORGANIZATION

The overall UV Sentry system and the specific UV Sentry scenario that was modeled in this thesis are described in Chapter II. The UV Sentry operational activities and Universal Naval Task List (UNTL) metrics pertinent to the communications system are then discussed and applied to the specific modeled scenario.

The major network design considerations are described in Chapter III. First, centralized network architectures are compared to mesh and ad-hoc architectures. Similarly, contention-based MAC protocols are compared with contention-free protocols. Finally, specific standards representing these different network considerations are discussed and a network standard is chosen for further modeling of the UV Sentry scenario.

The 802.16d and 802.16e standard's physical layer and MAC layer are described in Chapter IV. Additionally, the 802.16j transparent and nontransparent relay implementations are discussed in the context of their applicability to UV Sentry. Finally a modified 802.16j relay implementation is introduced as a more suitable relay method for UV Sentry.

The modeled functions for the application, MAC, and physical layers are described in Chapter V. Additionally, the network parameter choices for the model are explained. Then, the experiment design process and the factors selected for variation are described.

The model output data for both the uplink (UL) and downlink (DL) portions of the network are analyzed in Chapter VI. Packet delay, packet drop, overhead, and error metrics are analyzed based on their response to the variation of factors. Next, the impacts of these factors on network performance are characterized, and the best tested UV Sentry network configuration is recommended. Finally, this network configuration is demonstrated for a complete vehicle interception scenario in order to empirically evaluate the performance of the recommended network.

The significant findings of this thesis are summarized in Chapter VII. Additionally, recommendations are made on future research topics pertinent to the UV Sentry communications network.

Appendices comprise the functional hierarchy of the modeled communications system and source code for the model.

THIS PAGE INTENTIONALLY LEFT BLANK

II. UV SENTRY

The UV Sentry system will use unmanned vehicles to create "an autonomous capability for long-term, persistent and accurate surveillance, detection and engagement of threats that spans large geographical space and media [2]. The current method of utilizing unmanned vehicles is typically manifested through one or more people remotely controlling a single vehicle as with the Predator drone [11]. While this approach removes the human from the battlefield, it is still limited by the amount of manpower available, the ability of the human to maintain focus on the task at hand, and can be limited by the human inability to fuse large amounts of data from multiple vehicles that span across the multiple physical domains of sea, surface, undersea, and ground [2]. UV Sentry proposes to integrate these heterogeneous unmanned vehicles from multiple domains together into an automated system-of-systems. Through the effective use of these autonomous vehicles, UV Sentry will increase multi-domain persistence and awareness while at the same time reducing manpower dependence. At present time however, UV Sentry is not meant to completely replace the role of humans in surveillance but rather to be used in the coordination with other manned systems.

A. SYSTEM DESCRIPTION

UV Sentry connects multiple sensor systems and vehicles together into a network that is "minimally manned, highly autonomous force for long-term, persistent and accurate surveillance, detection and engagement of threats . . . [that] leverages parallel, disparate autonomous system technologies into a network-centric, system-of-systems architecture" as illustrated in Figure 1 [2]. UV Sentry is envisioned to accomplish a number of missions that are currently dominated by manpower intensive systems. These missions include maritime facility protection, counterdrug operations, riverine operations, maritime domain awareness, mine warfare, anti-submarine warfare, anti-surface warfare, and missile defense.

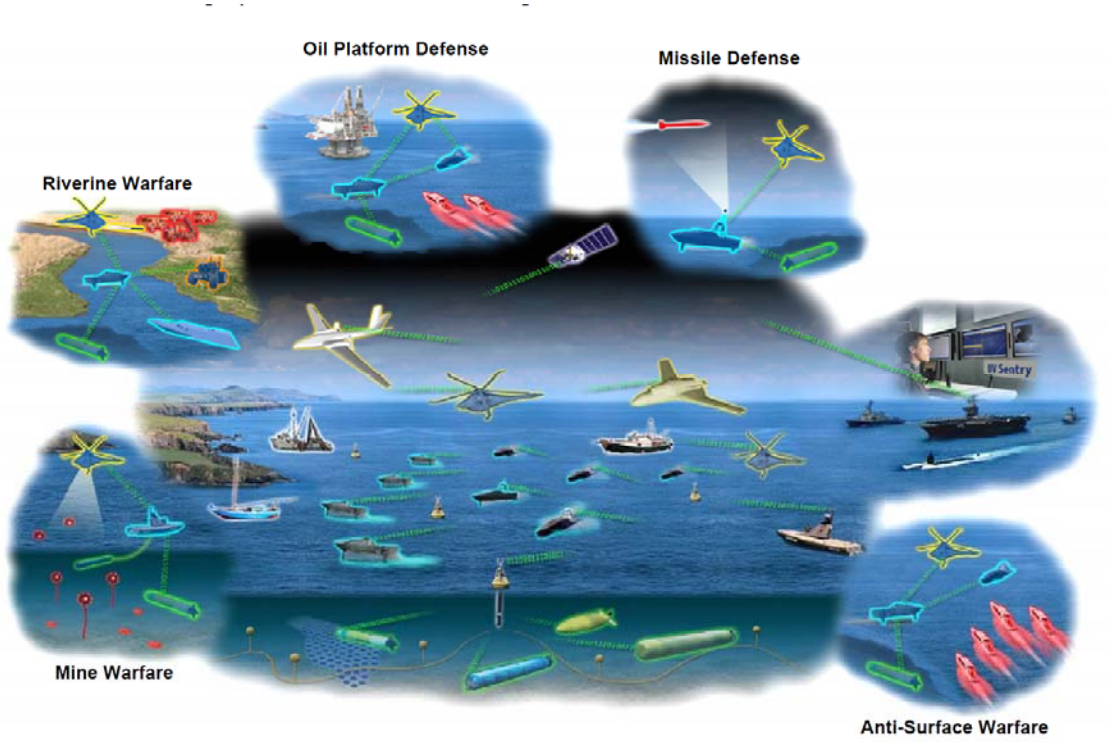

Figure 1. UV Sentry operational view 1 (OV-1) demonstrating the multiple potential missions that UV Sentry will conduct (From [2]).

The success of UV Sentry in large part will be dependent on the level of autonomy that the system developers are able to implement. This is illustrated by the four UV Sentry critical enablers: autonomous command and control; automated fusion of data from distributed, heterogeneous sensors; automated target "discernment;" and automated launch, recovery, and sustainment [2]. These enablers will allow the UV Sentry vehicles to gather dispersed data from sensors and compile a more complete view of the threat environment. Autonomously detecting a target and then acting on that knowledge independent of a human operator will both increase the system response time and disentangle the UV Sentry surveillance capacity from manpower restraints. A single operator can supervise several UV Sentry vehicles simultaneously. Furthermore, by autonomously launching, recovering, and sustaining itself, the manpower constraints on the system will be substantially reduced.

In the end, this move towards autonomy is meant to "wean [unmanned vehicles] from the bandwidth. Greater autonomy should be developed to reduce data requirements sent 'to' the [UV Sentry vehicle], and more advanced automated target recognition must be developed to reduce data requirements 'from' the [UV Sentry vehicle]." [2] Nevertheless, since this automation technology is still being developed, the UV Sentry communications network will need to bridge the gap between the first iterations of UV Sentry that will be more reliant on their human operators and the final iterations of UV Sentry that will be largely autonomous and more able to operate independent of human supervision.

The success of UV Sentry will significantly depend on the ability of its communications network to robustly connect vehicles. One of the goals for UV Sentry is to "be robust with respect to intermittent communications" and to perform functions "across a distributed system of unmanned platforms, without relying on a communications link back to the control station." [2] However, in the case that this level of automation is not reached in the first iterations of UV Sentry, the network must be capable of allowing a human operator to monitor the sensor data received from the UV Sentry vehicles and augment the automation as necessary. Accordingly, a communications network capable of bridging the gap between the two extremes of full human operator control and complete vehicle automation is the optimal choice for the first deployments of UV Sentry.

B. SCENARIO

In order to properly design a communications system for the UV Sentry, it is best to understand the UV Sentry system as a whole and then derive the communications needs based on the operational activities conducted in specific scenarios. As UV Sentry proves more capable of autonomous operation, the system will be deployed into successively more challenging environments with more challenging missions. However, for the first deployments and for the purposes of this thesis, a simple scenario will be considered. This simple scenario will focus primarily on the mission of protecting a fixed position maritime facility from hostile surface vehicles. While the communications

network design will consider more challenging scenarios and environments in the design choices to allow network scalability and flexibility, the modeling and analysis will be limited to the maritime facility protection scenario described below.

1. Mission

The protection of offshore oilfields and other fixed maritime facilities presents a significant security challenge. The offshore oilfields are characterized by multiple, individual oil platforms that potentially could be dispersed across thousands of square miles. The large scale damage that could occur due to an attack on one of these oil platforms is evidenced in the 2010 Gulf of Mexico oil spill, which caused British Petroleum to establish an $20 billion fund to cover the initial costs of the oil spill – a tragedy whose ultimate cost is still unknown [12]. While the British Petroleum oil spill was an accident, it underscores the vulnerability of these assets and the scale of damage that might occur if they were attacked. Furthermore, some of these maritime oil facilities are strategic level targets that could result in catastrophic economic or environmental damage if destroyed. For instance, the oil that follows through the Khawr al Amaya Oil Terminal and the al Bakr Oil Terminal in the northern Persian Gulf accounts for approximately 90% of Iraq's gross domestic product [13]. If a successful attack was made on one of these oil platforms, it would most likely cripple the Iraqi economy and would be a significant blow to Coalition stabilization efforts. For the purposes of this scenario, the maritime facility will be assumed to be an oil terminal. The protection of these offshore oil rigs and terminals is a critical mission in United States Central Command (CENTCOM) and United States African Command (AFRICOM), and for the purposes of this scenario, they will be considered the primary stakeholders.

The challenge of protecting such terminals from a hostile threat requires that they be protected from attacks from the domains of air, surface, and subsurface. The security solution should also be cross-domain and capable of executing the complete kill chain to include surveillance, tracking, threat discernment, engagement, and battle hit assessment. The scenario is further complicated due to the typically busy littoral environments which are generally characterized by high levels of white shipping and commercial fishing

vessels. This requires a distributed solution that is capable of tasking multiple sensors to conduct simultaneous and comprehensive surveillance of a large geographical area in order to quickly differentiate legitimate threats from the white background traffic.

2. Environment

The UV Sentry will need to be able to operate in a wide range of physical environmental conditions. However, for this scenario, the following conditions were chosen based on the UV Sentry Design Reference Mission [14]:

Sea State	<3
Water Temperature	95 F
Bathymetry	Max depth 90 meters, average depth 50 meters
Wind Speed	Low wind speeds (< 10 kts)
Visibility	Clear
Time of Day	Dawn

3. Threat

As discussed previously, a threat to the maritime facility could emerge from many domains. Furthermore, from each domain, the threat can take many different forms. From the air domain, the most probable threats include a suicide bomber in a civil aircraft or an armed UAV. From the surface domain, the most probable threats include manned attack boats, suicide boats, or armed unmanned surface vehicles (USV). From the undersea domain, the most probable threats include swimmers or armed unmanned underwater vehicles (UUV) [2]. Additionally, these threats might have access to sophisticated weapons such as missiles or torpedoes acquired through a state sponsor. It is assumed that the most likely result of an attack on an oil platform would be to temporarily disrupt oil production. However, the most dangerous result of such an attack would be to inflict enough damage to severely cripple the oil production capability of a

transfer facility. Thus, a system that is capable of quickly intercepting and identifying a threat would allow greater time to engage the threat before it closes within range of harming the oil platform.

4. UV Sentry Vehicles

In the complete scenario, the UV Sentry system utilizes a variety of multi-domain autonomous vehicles that are automatically launched, recovered and tasked from an operating platform such as the maritime facility itself or a Littoral Combat Ship (LCS) that may be positioned nearby. To provide radar surveillance of surface and air contacts in the vicinity, an elevated radar sensor can be used [15]. This sensor could be positioned on and operated from the defended platform itself or placed on a stationary vehicle such as an aerostat that would remain over the defended platform. USVs will provide a constant distributed presence around the defended platform for the purposes of surface surveillance, interception, identification, and tracking of possible surface threats. Additionally, the USVs will be able to provide a communications relay capability for other UV Sentry vehicles. The maximum speed of the modeled USVs is approximately 40 kts [16]. To provide undersea surveillance close to the defended platform, tethered ASW sensors and UUVs with acoustic sensors are used to detect underwater threats. Further away from the operating platform, Vertical Takeoff UAVs (VTUAV) with a dipping sonar capability can be used for surveillance of the undersea environment in zones further away from the defended platform. Additionally, the VTUAV and can also be tasked with providing communications relay capability and identification of possible surface threats. The maximum speed of the VTUAV is approximately 115 kts and the maximum endurance speed is approximately 50 kts [17]. "Data from these distributed multi-domain sensors is automatically fused to form a Common Operational Picture (COP), which provides UV Sentry and manned operators with situational awareness and the means to allocate assets and make engagement decisions." [15] The UV Sentry communications network must be able to facilitate the communication of these dispersed vehicles moving at high speeds in order to provide sufficient information to the human operator and also to the other UV Sentry vehicles in the network.

5. Ranges

To defend the fixed maritime facility, the UV Sentry scenario proposes three predetermined zones that extend outward from the maritime facility as shown in Figure 1. The outer zone is the surveillance zone set at 7 km; UV Sentry will autonomously detect and track all contacts inside this zone. The middle zone is the warning zone set at 3 km; UV Sentry will warn all unauthorized contacts in this zone that they are about to enter the exclusion zone. The inside zone is the exclusion zone set at 2 km; UV Sentry will interdict and possibly engage identified threats inside this zone.

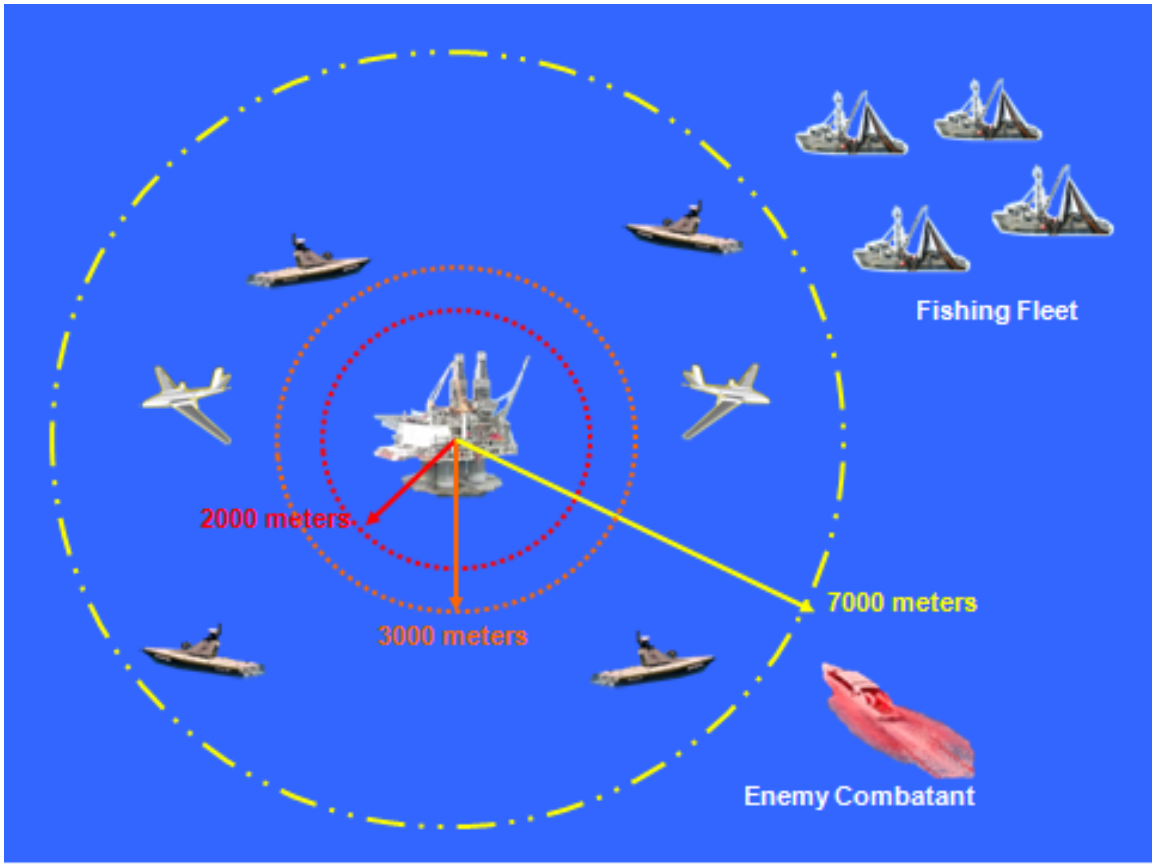

Figure 1. Modeled UV Sentry scenario vehicles and ranges (After [14]).

6. Scope

A detailed modeling and simulation of a communications network that can support the demands provided by the scenario above would be very complex. Because of this, assumptions concerning the scenario were made to simplify the model. To limit the scope of this thesis, the ability of the UV Sentry system to combat threats originating from the undersea and air domains were not evaluated. Here, the emphasis is solely on the design of a communications network that enables the vehicles tasked with the surface surveillance role to intercept threatening surface vessels.

The air surveillance role is best accomplished by a central radar system that would most likely be positioned on the operating platform. This operating platform would be the central node for UV Sentry. For the first UV Sentry deployments, countering an air attack would most likely be accomplished by point defenses originating from the operating platform and not from the distributed UV Sentry vehicles.

Communication with UV Sentry sensors and vehicles in the undersea domain was excluded due to the significant differences between providing a communications link to undersea assets when compared to providing a communications link to surface and airborne assets. UUVs may have the capability to come to the surface to transmit undersea warfare (USW) reports, and VTUAVs would have the ability to transmit USW reports attained from their acoustic sonar. However, due to the infrequent nature of these reports, a subsurface attack scenario was not believed to be the most stressing scenario for a wireless communications network. Therefore, the subsurface threat scenario is not considered to be the most demanding scenario in regards to the communications network.

A surface attack was chosen as the scenario that would be the most demanding for the UV Sentry communications network and will be the focus of this thesis. In response to the bombing attack on the *USS Cole*, the Office of Naval Research listed the Boghammar class boat and similar boats as possible threats in the littoral environment [18]. These boats can achieve top speeds of up to 46 kts [19]. Additionally, gunmen in speedboats have proven to be a common method of attack in the AFRICOM AOR. The

region has recently been plagued by militants in these speedboats raiding oil platforms [20], [21]. Therefore, this analysis will focus on a single speedboat as the threat to the protected oil platform.

Initially, the UV Sentry surveillance system will be able to autonomously detect, track, and identify vehicles as contacts of interest (COI). A wide variety of sensors may be installed on the UV Sentry vehicles to locate, track, and identify COIs. Once the COI is identified by UV Sentry, a human operator may want to be able to view the "high-resolution imagery of a contact . . . with a single mouse-click" in order to verify the UV Sentry's target identification [2]. To visually identify the COI, full motion video is preferred to still images because "full motion imagery provides event fidelity and seamless event progression, thus allowing an analyst to gauge not only three dimensions of still imagery, but also observe movement and scene changes over time." [22] Even though autonomous behavior recognition algorithms may be deployed in UV Sentry, the UV Sentry operator may desire to watch the imagery in order to more fully interpret the intent of the surface vessel occupants. This can be done by gauging their response to interception by a USV or by perhaps having momentary glimpses at weapons onboard a vehicle that may not be apparent from still images. Enabling UV Sentry to transmit NRT video back to a human operator will also enable later iterations of UV Sentry to engage hostile targets. It is this NRT video capability that allows the Predator UAV "to employ weapons from a UAV; a capability only possible because of the NRT aspect of the sensor that allow[s] the platform to detect and identify a target in a manner that [meet] the Rules of Engagement (ROE) constraints for weapons release." [11] Although NRT video enables the operator to potentially fulfill the identification requirements for hostile vehicles according ROE, the communications necessary for the actual engagement of vehicles will not modeled.

7. Phases

The UV Sentry system details several phases of rising escalation based on the perceived threat from a COI. It is useful to detail the responses of the UV Sentry vehicles

in order to assess their communications needs for different phases and to identify points at which the communications network might have the greatest demand placed on them.

a. Detection and Identification Phase

Throughout the detection and identification phase, multiple UV Sentry vehicles will monitor the surveillance zone. Also, the radar positioned at the central maritime facility scans the region for air and surface contacts [15]. The UV Sentry system monitors all contacts within the surveillance area as COIs. Furthermore, all contacts who are displaying unusual patterns (such as speed increases or a lack an Automatic Identification System or Identification Friend or Foe) within the vicinity of the surveillance area are also labeled as COIs. UV Sentry vehicles are forward positioned to patrol the surveillance area and interdict these COIs. These forward deployed UV Sentry vehicles can transmit preprocessed sensor information that summarizes what they have observed so that it can be fused with other UV Sentry sensor data. This information is then distributed amongst all the network nodes to form the COP. Also, the human operator can also request to view more detailed information if he or she desires to see the direct output of a UV Sentry vehicle's onboard sensors.

b. Intercept and Engage Phase

When any unauthorized surface contact crosses within the surveillance zone, UV Sentry automatically tasks a vehicle to interdict that contact [15]. Initially, the identity of the contact may initially be unknown. However, as a UV Sentry vehicle gets closer to the target, the UV Sentry vehicle will be able to identify the target by using onboard automated target recognition software. The target detection and identification is accomplished automatically by UV Sentry; however, UV Sentry's decision and sensor information are available to the human operator for monitoring. Also, at any time the automated UV Sentry commands may be overridden, and the human operator would be able to command the vehicles manually. As the unauthorized vehicle approaches the warning zone, the UV Sentry vehicle will attempt to overtly warn the unauthorized vehicle. However, if the unauthorized vehicle continues to enter the engagement zone, it

may be classified as hostile target and engaged by a UV Sentry vehicle or weapons system external to UV Sentry. For the purposes of this scenario, the engagement phase was not modeled. However, as described in [2], a direct authorization command from a human decision-maker would be required for any use of lethal force. After the target has been engaged, the UV Sentry vehicles will also conduct battle damage assessment to evaluate if a reengagement is necessary.

C. OPERATIONAL ACTIVITIES

In order to accomplish the mission as specified in the above scenario, UV Sentry would have to accomplish several functions. A full functional breakdown of UV Sentry was created by WBB Consulting in the *Unmanned Vehicle Sentry Architecture* report [23]. The functions listed in the report cover the full spectrum of UV Sentry operations, but only a subset of the functions directly relate to the topic at hand, namely UV Sentry communications. These functions are referred to as "operational activities" in the report and can be broadly categorized into three overarching operational activities of "sense," "command and control," and "communicate." These operational activities are further detailed in the following section.

1. Sensors

A wide variety of possible sensors could be incorporated into the UV Sentry system as illustrated in Table 1, but not all of them are included in the scenario model. The sensors that were chosen to be modeled on the scenario are the sensors that are currently widely deployed on manned and unmanned military platforms. To collect imagery, the UV Sentry system lists three possible options: "Collect Hyperspectral Imagery" (1.5.3), "Collect EO, IR, Multi-Spectral Imagery" (1.5.4), and "Collect Laser-based Imagery" (1.5.5). Of these imagery options, "Collect EO, IR, Multi-Spectral Imagery" was chosen as the type of imagery sensor deployed on the UV Sentry vehicles. This is because of the widespread deployment of electro-optical/infrared (EO/IR) systems on existing platforms. While they may not be as precise as the other choices, they have been shown to be good enough to allow remote engagement [11]. For distant contact

detection and tracking, the operational activity “Collect RF Data” (1.5.6) was chosen. This was due to the need of the vehicles to be able to surveil a large area looking for contacts, and radar systems are widely deployed on current platforms for this purpose.

There were several sensors not included in this model because they were not pertinent to the scenario. The operational activities of “Collect CBRN Data” (1.5.1) and “Collect Explosives Data” (1.5.7) were excluded because these sensors are envisioned to require very little bandwidth. This sensor data could possibly be included in a message the vehicle sends to report on other status indicators such as fuel level, position, etc. The operational activity “Collect Environmental Data” (1.5.2) was excluded because it was assumed that the operating platform would be able to collect this information for the geographical region. Operational activates “Collect Hydrographical Data” (1.5.8) “Collect on Underwater Objects” (1.5.9) and “Collect SIGINT Data” (1.5.10) were excluded because the scenario was designed to only protect the maritime facility against unsophisticated surface threats and not underwater targets.

Table 1. UV Sentry sensor operational activities (From [23]).

Operational Activity		**Description**
1.5.1	Collect CBRN Data	Collection of information searching for indicators of Chemical, Biological, Radiological, or Nuclear (CBRN) contamination of an operating area.
1.5.2	Collect Environmental Data	To collect meteorological, oceanographic, and space environmental factors.
1.5.3	Collect Hyperspectral Imagery	Term used to describe the imagery derived from subdividing the electromagnetic spectrum into very narrow bandwidths. These narrow bandwidths may be combined with or subtracted from each other in various ways to form images useful in precise terrain or target analysis.
1.5.4	Collect EO, IR, Multi-Spectral Imagery	To collect electro-optical (EO) or infrared (IR) intelligence data.
1.5.5	Collect Laser-based Imagery	Collect laser-based imagery (LADAR/LIDAR).
1.5.6	Collect RF Data	Execute commands that operate sensors that radiate radio frequency (RF) in order to get a return
1.5.7	Collect Explosives Data	Collect with the intent of discovering explosives in target objects.
1.5.8	Collect Hydrographical Data	Reconnaissance of an area of water to determine depths, beach gradients, the nature of the bottom, and the location of coral reefs, rocks, shoals, and manmade obstacles.
1.5.9	Collect on Underwater Objects	Collect using various means with the intent of discovering underwater objects (mines, submersibles, semi-submersibles).
1.5.10	Collect SIGINT Data	Collect signals intelligence (SIGINT) data.

The sensor data that is collected by the vehicles is not automatically transmitted to the other vehicles or to the human operator. Instead, vehicles will autonomously detect a target, use target recognition software to identify it, and watch for detected anomalous behaviors [2]. Based on the collected data, UV Sentry will attempt to infer if the contact has a hostile intent using behavior recognition algorithms. All the UV Sentry sensor information will be distilled into a COP message that will be transmitted to other vehicles

as well as the operating platform. The human operator will be able to subscribe to a more detailed view of the sensor data, at which point the actual radar or EO/IR video data will be transmitted to the operating platform for observation.

2. Command and Control

The command and control functions for UV Sentry listed in Table 2 are envisioned to be largely executed autonomously. The vehicles will be able to collaborate amongst themselves and perform autonomous tasking of each other based on the vehicle's self-generated COP. These tasks will be prioritized and allocated based on the human provided mission goals, constraints, and commander's intent. This mode of control is encompassed by the operational activity "Control Vehicles" (3.6). However, at any time, the human will be able to monitor the system activity, receive high level alerts on the status of the UV Sentry vehicles and COIs, and receive detailed sensor data of COIs when requested. These monitoring functions fall under the operational activity "Monitor Active Vehicles (Steady State)" (3.4). At this time, the human operator can switch from the autonomous control to a manual method of control where he directs the vehicles movements through the "Control Vehicles" (3.6) operational activity or, more specifically, direct the vehicles movements through the "Coordinate with Manned Assets" (3.7) or "Reapportion Assets" (3.9) operational activities. Also, vehicles can join or leave the local UV Sentry system for reasons such as being tasked to another UV Sentry system or for local refueling or maintenance operations. The "Establish Link" (3.3) and "Execute Vehicle Handoff" (3.11) operational activities account for the entering and leaving of these vehicles. Nevertheless, for the simulations conducted, no vehicles will be modeled as entering or leaving the communications network.

Table 2. UV Sentry command and control operational activities (From [23]).

Operational Activity		Description
3.3	Establish Link	Establish communications links between Unmanned Vehicle Sentry Command and Control assets and UV Sentry vehicles.
3.4	Monitor Active Vehicles (Steady State)	To perform the standing mission in accordance with the current mission plan. Once a plan has been developed, or deviations to the current plan have been decided and acted upon, steady state ensues upon the assets beginning to execute that plan.
3.6	Control Vehicles	Control vehicles under the auspices of Unmanned Vehicle Sentry.
3.7	Reapportion Assets	Upon decision to reapportion assets in response to a contingency, mission orders are distributed to the unmanned vehicle participants, and the mission continues.
3.9	Coordinate with Manned Assets	To coordinate with manned assets in the area in order to perform such operations as boarding, search-and-seizure, tactical reconnaissance, or other tasks better suited to the manned assets in the area.
3.11	Execute Vehicle Handoff	To receive control of, or cede control of a vehicle to/from another agency.

3. Communications

The above-mentioned sensor and control operational activities will generate communications operational activities as the information is transmitted between UV Sentry nodes. The operational activities that dictate the communication requirements for the UV Sentry network are listed in Table 3. Sensor data can be transmitted in distilled form using a COP message via the "Track Targets" (1.9) operational activity or it can be transmitted in the more detailed form of imagery or video through the "Collect Sensor Data" (1.5) operational activity. When the "Autonomously Navigate Vehicle" (1.10) operational activity is being executed, the vehicles coordinate amongst themselves via COP messages to determine their movements. However, a human can intervene using the "Navigate Using Route Planning" (1.11) and "Respond to Navigation Commands" (1.12) operational activities to manually update or control the movement of the UV Sentry Vehicles. UV Sentry provides an operational activity for vehicles to exit the

network via "Autonomously Recover Vehicle" (1.15) and an operational activity for vehicles to enter the network via "Join Mission Networks" (1.16). However, for the simulations conducted, no vehicles will be modeled as entering or leaving the communications network. The "Relay Communications" (1.17) operational activity accommodates the distributed UV Sentry topography where not all UV Sentry nodes are assumed to be within range of the central operating node. Finally, the "Report Vehicle Status" operational activity describes the communication necessary for vehicles to report their navigational parameters (location, speed, heading, etc.) and subsystem parameters (fuel level, engine temperature, etc.).

Table 3. UV Sentry communication operational activities (From [23]).

Operational Activity		**Description**
1.5	Collect Sensor Data	Collect on a target area using various sensors, with the intent of producing products for analysis.
1.9	Track Targets	To monitor or follow the course, location, and/or status of a Target of Interest.
1.10	Autonomously Navigate Vehicle	Navigate vehicle autonomously using a ruleset provided during mission planning.
1.11	Navigate Using Route Plan	Navigate using route plan provided by external entity.
1.12	Respond to Navigation Commands	Execute navigation commands issued by a valid controlling organization.
1.15	Autonomously Recover Vehicle	Recover vehicle autonomously, without operator involvement. The vehicle will autonomously dock or land itself based upon information given to it when it requests clearance to return to the launch vessel.
1.16	Join Mission Networks	Join networks as identified in the communications plan embedded in the mission plan for the vehicle.
1.17	Relay Communications	Relay communications from one asset to another.
1.18	Report Vehicle Status	Monitor and report vehicle and vehicle subsystem status.

The above communications operational activities can categorized into four distinct message types for the purposes of modeling the network traffic. The first

message type is the control information transmitted to a vehicle. If in automated mode, this information will be a constant stream of COP and collaboration messages to facilitate the distributed control of the UV Sentry system. If in the control mode, this information will be more directive in nature as the operator directs the vehicle to accomplish certain tasks, such as maneuver to a specific waypoint or investigate a specific contact. The second message type is the control information transmitted by a vehicle. This information will be a constant stream of data concerning the vehicles navigation status (position, heading, etc.), equipment status (fuel level, engine temperature, etc.), and contact status (distilled sensor analysis of nearby contacts). The third message type is the radar sensor data from the vehicle that can be requested for analysis by the human operator. The fourth message type is the EO/IR sensor data from the vehicle that can be requested for analysis by the human operator. From this initial analysis of the different message types transmitted in the network, we can identify the messages that the UV Sentry vehicles will transmit and receive in the network. A detailed description of how these messages are modeled is discussed in Chapter V.

D. METRICS

In order to measure the effectiveness of the UV Sentry communications network in accomplishing the above operational activities, the performance of the communications network must be measured using common metrics. The Universal Naval Task List (UNTL) "provides a common language that commanders can use to document their command warfighting requirements as mission essential tasks." [24] The tasks listed in the UNTL level are tactical and provide the level of detail necessary for the performance measurement of specific systems. For each task, a list of metrics is also provided as a basis of system assessment. The UNTL tasks and metrics will serve as a guide for measuring the performance of the UV Sentry communications network.

The UNTL provides two tasks with the level of detail necessary for the evaluation of a communications network. These tasks are listed in Table 4 and are accompanied by the task description and metrics.

Table 4. UNTL communication tasks and metrics (From [24]).

NTA 5.1.1.1.1 Provide Internal Communications
To send and receive information required for own unit operations and to provide tactical information through the use of internal communication systems. (JP 3-0, 6-0, 6-02, NDP 6, Class Combat Systems Doctrine (CNSL/CNSP INST C3516 Series))
M1 Percent of time, desired communications path available.
M2 Minutes lag between commander's common picture of the battlespace and real world.
M3 Percent link data efficiency.
NTA 5.1.1.1.2.2 Relay Communications
To pass information which cannot reach its targeted audience directly. This includes the use of aircraft for tactical relay. (JP 3-0, 6-0, 6-02, NDP 6)
M1 Number messages relayed.
M2 Minutes to relay required messages.
M3 Percent correct messages received.

The first task "Provide Internal Communications" (NTA 5.1.1.1.1) is measured by three metrics and describes the communications of one UV Sentry vehicle to another. The first metric of "percent of time, desired communications path available" will not be evaluated in this thesis because of the limited geographical nature of the scenario. For the purpose of this analysis, we assume that all vehicles are interconnected either directly or through a relay and are free to maneuver as necessary to sense, receive and transmit based on the vehicle's individual tasking. The second metric of "minutes lag between commander's common picture of the battlespace and real world" is pertinent to the modeled communications network. In the scenario outlined here, the average and maximum delay of messages transmitted will be measured (in milliseconds and not minutes) to compare the simulated communications networks. Moreover, messages may be dropped because the network has been overloaded, and the delayed messages are no longer relevant. The third metric of "percent link data efficiency" refers to the message overhead necessary to transmit the message. To evaluate the communications network, we will examine the overhead necessary to transmit the UV Sentry messages.

The second task "Relay Communications" (NTA 5.1.1.1.2.2) is also measured by three metrics and describes the communications of UV Sentry nodes through relays. The

first metric of "number messages relayed" will not be evaluated because the actual number of messages that are relayed will vary based on the network topology and sensor utilization. The second metric of "minutes to relay required messages" will be evaluated in the same way as the above metric of "minutes lag between commander's common picture of the battlespace and real world." The only difference will be that when a relay is used to transmit a message, it will be recorded as such in order to differentiate it from unrelayed messages. The third metric of "percent correct messages received" will be evaluated based on the percentage of correct control, status, and sensor messages that were received for both relayed and unrelayed messages. This metric is directly related to the error rates associated with the network communications link, and we will examine the reliability of these links in addition to evaluating the percentage of messages correctly received.

E. MODELED SYSTEM

For the network simulation model, we will simulate six UV Sentry vehicles in addition to the central operational platform that will be collocated with the fixed maritime facility. Four of the vehicles will be USVs that will be prepositioned around the fixed maritime facility for surveillance. The USVs will have onboard radars that will be used to locate and track local contacts. Additionally, two VTUAVs will be airborne and patrolling at a maximum endurance speed. Both the VTUAVs and the USVs will have onboard EO/IR or FLIR cameras with the capability of onboard autonomous target recognition and the ability to transmit NRT FLIR video back to the operation platform. The VTUAVs and the USVs will also be capable of providing communications relay for other UV Sentry vehicles that are further away from the operating platform. In the modeled scenario, all vehicles will transmit their portion of the COP in order that a fused COP is available for all vehicles. All vehicles will also transmit their current navigation and equipment status. As previously stated, no UUVs or underwater sensors were included in the simulation. A summary of the modeled system is shown in Table 5.

Table 5. UV Sentry modeled scenario summary.

Scenario	Fixed Maritime Facility Protection
Operating Platform Location	Maritime Facility
Mission	Oil Platform Defense
Threat	Armed Speedboat
Number of Threats	1
Threat Max Speed	46 kts
Number of USVs	4
USV Max Speed	40 kts
Number of VTUAVs	2
VTUAV Max Endurance Speed	~50 kts
VTUAV Max Speed	115 kts
Vehicle Video Sensor	EO/IR (FLIR)
Vehicle RF Sensor	Radar
Relay Enabled	Yes
Target Engagement	No
Surveillance Zone	7000 m
Warning Zone	3000 m
Engagement Zone	2000 m

III. NETWORK DESIGN CONSIDERATIONS

There are multitudes of wireless communication networks types, and each is designed to meet a specific communications requirement. To choose the best communications network for UV Sentry, we examine the different network architecture types and then study in more detail the actual mechanisms that enable the coordinated communication of the individual network nodes. We then compare the available options with the requirements of UV Sentry to best choose a communications network design for the UV Sentry system.

A. NETWORK ARCHITECTURE

To choose the best communications network for UV Sentry, we compare two major network architecture types: centralized and mesh/ad-hoc. Centralized networks have a distinct hierarchy structure, whereas the mesh or ad-hoc networks have a more distributed control structure. The choice of which type to use depends on the type of demands placed on the network.

1. Central Control Network Architecture

Cell phones, the IEEE 802.11, and the IEEE 802.16 communications networks have a similar central control network architecture; multiple subscriber stations (SS) are connected to and controlled by a single BS. All the nodes in the network share the same physical wireless medium, and the BS centrally regulates access to that medium via a contention-based or schedule-based protocol. With a tightly controlled regulation of who can access the channel, these networks can attain high throughput at higher utilization and can guarantee QoS parameters [25]. Nevertheless, this architecture structure can only cover geographical distances that do not exceed the wireless transmission distances of the BS and SSs. This BS coverage region is sometimes referred to as a cell. To solve coverage problem, the BS also serves as the access point to the exterior network, so that it

controls access to a larger network. In this manner, multiple cells can be connected together by connecting the cell BSs. In so doing, a large geographical area can be covered by a wireless network.

Another method of extending coverage with the central network architecture is to make use of relays. These relays repeat the messages received to other network nodes that are not within range of the original transmission. While these relays may enable the network to transmit beyond line-of-sight (LOS) distances using multiple hops from one relay to the next, they do so at the cost of lower network throughput [26]. The capabilities of these relay stations (RS) vary from simple repeaters to radios that are similar to the BS. These advanced RS act similar to BS in their responsibility for regulating the medium access control of their subservient SS [26]. If the SS are mobile and constantly changing the network topology, these advanced RS can also adjust to the SS movement by transitioning the connection of the SS from one RS to another RS or back to direct communication with the BS.

2. Mesh and Mobile Ad-Hoc Networks

Mesh and ad-hoc networks lack the centralized control structure that characterizes the cellular and multi-hop networks listed above. The mesh and ad-hoc nodes are dynamically self-organized and self-configuring [27]. Because of their distributed nature, mesh networks can cover a large geographical area. There generally is not a BS/SS hierarchy where the BS controls access to the external network. Instead, multiple individual network nodes can be connected to an external network through a wired or wireless connection. However, these nodes can also simply exist with the sole purpose of being a relay point between other nodes. While mesh networks are generally fixed, ad-hoc networks support node mobility so the topology of the network changes considerably more often. Despite these advantages, mesh and ad-hoc networks are characterized by a low end-to-end throughput and difficulty in guaranteeing QoS at high network utilization [27], [28].

B. MEDIUM ACCESS CONTROL LAYER

Due to limited physical medium resources, access to the network's wireless communications channel must follow a given protocol. The MAC protocols divide up the communications channel in order to allow multiple nodes on the network to be able to share the medium resources. Common methods of dividing up the physical medium are: frequency, time, space, and code. Accordingly, these methods are classified as frequency-division multiple access (FDMA), time-division multiple access (TDMA), space-division multiple access (SDMA), and code-division multiple access (CDMA). By dividing up the physical medium's available frequency spectrum into smaller frequency channels, these individual channels can then be allocated to different users. By dividing up the physical medium into time slots, separate time slots can be utilized by different users. By using multiple antennas that make separate and simultaneous transmissions, these separate transmissions in the same frequency can be distinguished through SDMA. By using a unique code to encode each transmission, simultaneous transmissions in the same frequency can be distinguished through CDMA. Furthermore, protocols can be constructed to use a combination of the above methods.

Current wireless medium access protocols generally fall into two broad categories: "scheduled" and "contention-based." Scheduled protocols provide high levels of throughput when the communications network is under high demands. Nevertheless, this level of performance has the cost of higher levels of overhead and delay [29]. In contrast, contention-based approaches provide lower delays when the network is under lower demands. However, if the network utilization increases past a certain threshold, the network performance becomes significantly slower [25], [30].

1. Contention-Based Wireless Medium Access Control Protocols

Contention-based schemes have evolved and become increasingly sophisticated to deliver higher levels of throughput under increased network demands. The first wireless medium access protocols was the ALOHA protocol [30]. This simple protocol allowed a node to transmit a data packet as soon as it was received. If there was no acknowledgement received or a negative acknowledgement received, the node

retransmitted the packet after a random delay. Carrier sense multiple access (CSMA) improved on this protocol by having the node with data to transmit "listen" or sense the physical medium beforehand to detect if another node in the network was transmitting [25]. If the medium was not busy, the node would transmit the data packet. However, if the medium was busy, then the node would wait a given time interval before sensing the medium again.

Nevertheless, both of these protocols suffered from the hidden node problem and the exposed node problem. The hidden node problem occurs when then two nodes (nodes A and C) are within reception range of a third node (node B), but they are not within reception range of each other as in Figure 2. Because they are out of range of each other, they may attempt to transmit to the third node simultaneously and interfere with each other. The exposed node problem occurs when there are four nodes as shown in Figure 3. In this case, node B desires to transmit to node A and node C desires to transmit to node D. However, while node A is out of range of node C and node D is out of range of node B, simultaneous transmissions by node B and node C is prohibited.

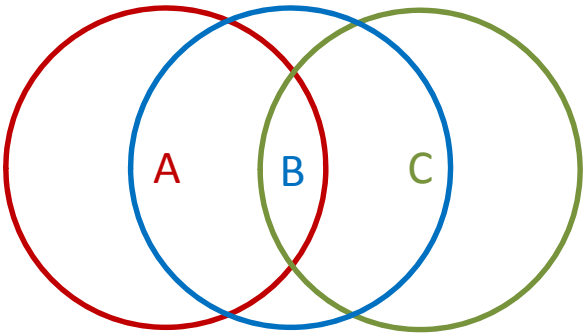

Figure 2. Hidden node problem where node A and C are hidden from each other.

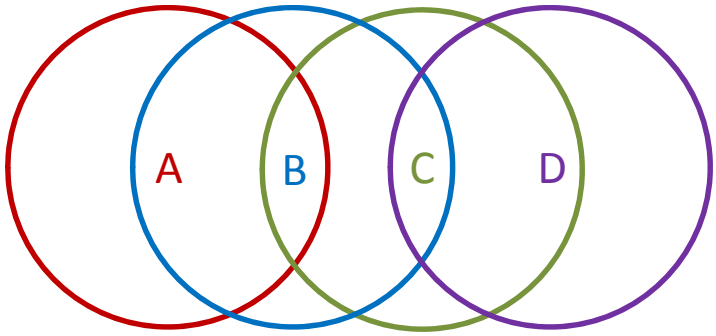

Figure 3. Exposed node problem where node A is out of range of node C and node D is out of range of node B.

To overcome the hidden node and exposed node problem, the Multiple Access with Collision Avoidance protocol was developed [31]. After a node has "listened" to ensure that the channel is unused, but before it sends its data, it first sends a Request-to-Send (RTS) packet to the receiver. The receiver replies by sending a Clear-to-Send (CTS) packet back to the transmitter. The transmitter then sends the longer data packet to the receiver. Additionally, if another node has a packet to send but hears a RTS transmission, it then listens for a CTS transmission. If it hears the CTS, it then waits until the transmission of the packet. If no CTS is heard, it assumes there is an exposed node (the other receiver node is out of range), and the node transmits its data. Instead, if a CTS is heard but not a RTS, it assumes that there is a hidden node and waits the length of time dictated by the CTS message.

The IEEE 802.11 standard [32] is a widely used specification that combines a contention-based access scheme with a contention-free scheme. For the contention-based portion, it relies on a version of CSMA that detects if another node is transmitting. If the medium is unused for a given length of time, it transmits data. If the medium does become used, the node waits until the medium becomes idle again and then waits an additional random length of time before it attempts a transmission. In addition to this scheme, there is a contention-free polling scheme that gives priority to specific nodes in order to deliver a specific quality-of-service to those nodes. This is accomplished by allowing these higher priority nodes to access the medium before the mandatory wait has elapsed for the lower priority nodes to contend for the medium. With 20 MHz of

bandwidth, IEEE 802.11g can provide up to 54 Mbps connections. Nevertheless, the 802.11 standard was designed for short-range wireless connections, and even with high power transmitters it can only support users up to approximately 300 meters from the wireless access point [10].

2. Contention-Free Wireless Medium Access Control Protocols

The collisions that occur when multiple nodes try to transmit at the same time and the delays that are associated with the nodes being "polite" and letting other nodes have a chance at transmitting on the physical medium allow a significant amount of the channel to be wasted. Instead of letting the individual nodes each contend for the channel, contention-free wireless MAC protocols schedule the medium's use. However, the scheduling messages create additional overhead traffic in the network. The medium can be scheduled by a centralized node, or the scheduling responsibilities can be distributed through the system. While centralized approaches are characterized as being efficient, they also face scalability issues. A distributed control may be less efficient but may be more scalable and allow for a more rapid response to changes in the network. Also, a network that updates the schedule more often will have higher overhead but be more responsive to changes in traffic demand when compared to a network that updates its schedule less frequently. In a cellular network framework, all scheduling is accomplished by the BS that allocates channel resources to the SS. The BS polls the SS periodically to determine an updated schedule for channel allocation based on negotiated QoS parameters and quantity of traffic agreements between the BS and the SS.

3G cellular systems are widely deployed to provide broadband wireless access to cell phone subscribers. A common worldwide 3G technology is the Universal Mobile Telephone System (UMTS) that uses the High Speed Packet Access (HSPA) standard to deliver high data rate transmissions between the BS and SS. Similar to the HSPA standard is the 1x Evolution-Data Optimized (EV-DO) standard. Both these standards are capable of delivering high data rate voice and video application data with low latency due to their scheduled protocol that allows QoS negotiation capability [33]. However, their bandwidth capability is not scalable and is set to a maximum of 14.1 Mbps for

HSPA and 4.9 Mbps for EV-DO [10]. Additionally, their maximum range capability is only about 5 km, and they do not have a relay capability specified in their protocol.

The original IEEE 802.16 standard also relies heavily on centralized scheduling for channel resource allocation [34]. The 802.16 MAC was designed to support very high peak bit rates and to meet QoS requirements such as minimum sustained rate, jitter, and latency [33], [10]. Additionally, the 802.16j amendment allows for a distributed MAC scheduling when using relays [35]. The network bandwidth is scalable from 1.25 MHz to 20 MHz so that varying network capabilities can be deployed to meet the specific needs of a network. Accordingly, it can currently supply data rates up to 75 Mbps. The standard 802.16 MAC protocol that supports 5 millisecond frames has a coverage range of up to 12.8 km, and longer ranges can be supported for nonstandard shorter frames [10].

Many of the coming fourth generation (4G) cell phones will employ a new standard called Long Term Evolution (LTE). This standard is planned to replace the current 3G infrastructure and will be able to support UL data rates of up to 50 Mbps [10]. Its underlying technology closely mirrors the IEEE 802.16 standard. However, the standard is not as mature as 802.16 and has not currently been deployed. Accordingly, it was not considered as a possible candidate for the UV Sentry network.

Lastly, Harris Communications advertises the SeaLancet RT-19944/U IP Network Radio that supports a 54 Mbps link rate radio [36]. This data rate equates to 32 Mbps of user data throughput at 5 nautical miles (9.2 km) [37]. In this direct mode, Harris also advertises that the radio can transmit to distances of up to 150 miles (at up to 8 Mbps of user data throughput). Additionally, it supports mesh mode, but the mesh mode is suspected to have significantly reduced performance due to the limitations of mesh networks previously discussed. Similar to the 802.16 standard, it uses the orthogonal frequency-division multiplexing (OFDM) physical waveform. Nevertheless, because the standard is proprietary, this analysis cannot discuss the SeaLancet as a possible candidate for the UV Sentry network.

C. UV SENTRY NETWORK CHOICE

Given the above considerations and the stated needs for the UV Sentry communications network, the IEEE 802.16 standard was chosen as the optimal standard for UV Sentry. The centralized cellular type of architecture was chosen over the distributed mesh and mobile ad-hoc network architectures. Even though the wide area coverage and self-healing nature of the networks is appealing, this architecture was not chosen due to its tendency towards throughput bottlenecks and latency [28], [27]. Furthermore, we believe that the use of relays can be used to extend the network coverage to meet UV Sentry requirements. The critical parameters of the most widely accepted wireless network standards are compared in Table 6: 802.11, 3G cellphone, and 802.16. The 802.16 protocol shows a greater capability in every category. It has the ability to provide a large amount of data throughput and uses its BW efficiently using the highly efficient OFDM waveform and a scheduled MAC. Moreover, it has a transmission distance that exceeds the requirements of the UV Sentry scenario and has the ability to use relay nodes to extend its range. Because of those features, the IEEE 802.16 standard was chosen as the network standard the UV Sentry communications network model.

Table 6. Summary of considered centralized network types where green denotes desirable traits, yellow denotes marginal traits, and red denotes undesirable traits (After [10]).

	802.16e	**HSPA**	**1x EV-DO**	**802.11 a/g**
Max Data Rate (Mbps)	75	14.4	4.9	54 (a)
Adjusted Max Data Rate (Mbps)	75	14.1	4.9	25
Max BW (MHz)	20	5	1.25	20
Data Rate/BW	3.75	2.88	2.72	1.25
Coverage (km)	12.8+	5	5	0.3
Scalability	Yes	No	No	No
MAC type	Scheduled	Scheduled	Scheduled	Contention
Relay Capability	Yes	No	No	Yes
Multiplexing	TDM/OFDMA	TDM/CDMA	TDM/CDMA	CSMA
(a) Due to inefficient contention MAC, 54 Mbps translates to 20 to 25 Mbps throughput.				

IV. IEEE 802.16 STANDARD

The IEEE 802.16 standard governs the physical layers and MAC layers for wireless networks. Worldwide Interoperability for Microwave Access (WiMAX) is an industry consortium implementation of the 802.16 standard, and the two terms will be considered synonymous for the purposes of this thesis. At the physical layer, they control how the available resources are divided up in both frequency and time. At the MAC layer resources are allocated to different users based on the parameters that the BS and the SS agree on. The standard has evolved since its initial introduction. The 802.16d is sometimes called "fixed WiMAX" in the industry implementation because it provides broadband wireless coverage to SS that are assumed to not be moving. To accommodate mobile users, the modified 802.16e standard (also referred to as "mobile WiMAX" in the industry implementation) was introduced. However, neither of the above standards accommodates relay capabilities. To resolve this, the 802.16j amendment to the 802.16e standard was released. We will discuss these standards in more detail and explain how they are implemented in the UV Sentry network in the following sections.

A. PHYSICAL LAYER

At the physical layer, the IEEE 802.16d standard makes use of OFDM. OFDM is similar to FDMA, except that instead of using nonoverlapping frequency bands for transmissions, OFDM divides the available spectrum channel into several independent sub-carriers. "This is achieved by making all the sub-carriers orthogonal to one another, preventing interference between the closely spaced sub-carriers. In an OFDM signal, all the orthogonal subcarriers are transmitted simultaneously. Orthogonally is achieved by making the peak of each sub-carrier signal coincide with the nulls of other signals where the result is a perfectly aligned and spaced subcarrier signal." [38] A single subcarrier is shown in the frequency domain in Figure 4, and the OFDM overlay of multiple aligned subcarriers is shown in Figure 5.

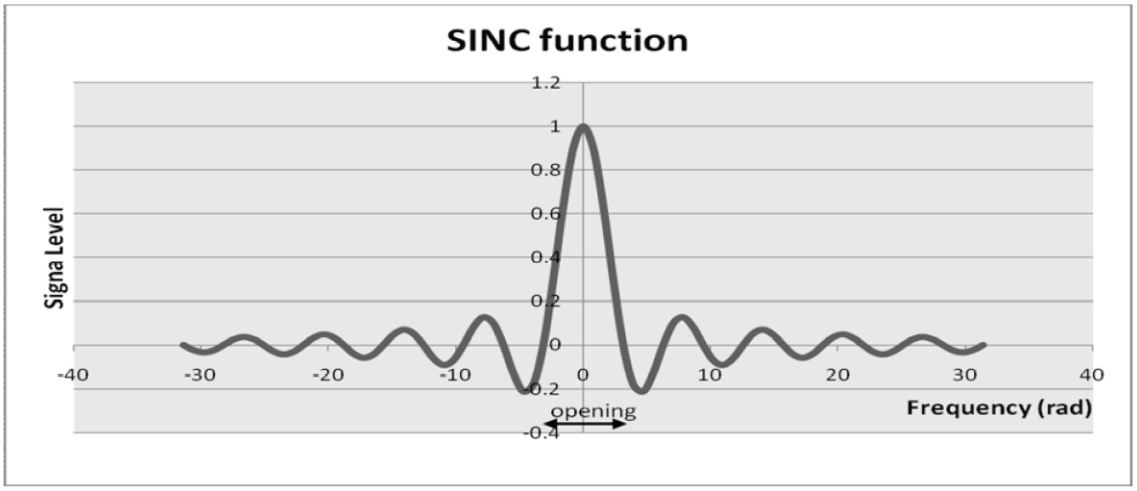

Figure 4. Single OFDM subcarrier in frequency domain (From [39]).

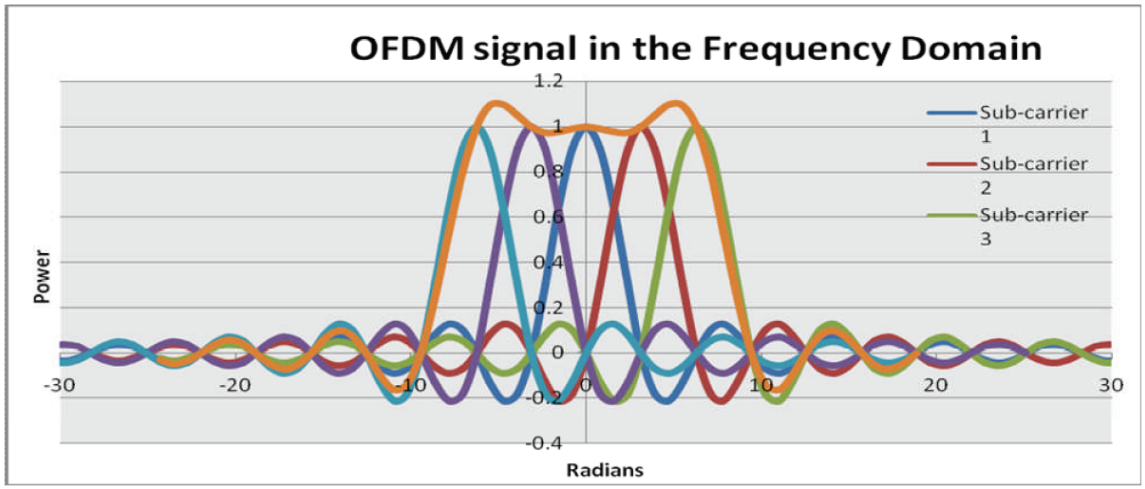

Figure 5. Multiple overlaid OFDM subcarriers in frequency domain (From [39]).

When viewed in the time domain, a succession of the above OFDM signals appear as in Figure 6, where each subcarrier can be associated with a specific frequency. However, the subcarrier transmission signal changes after a specific period of time. The period where a subcarrier transmission is constant for a specific period of time is known as a symbol and is the basic unit of the OFDM signal. Each box in Figure 6 represents a unique OFDM symbol.

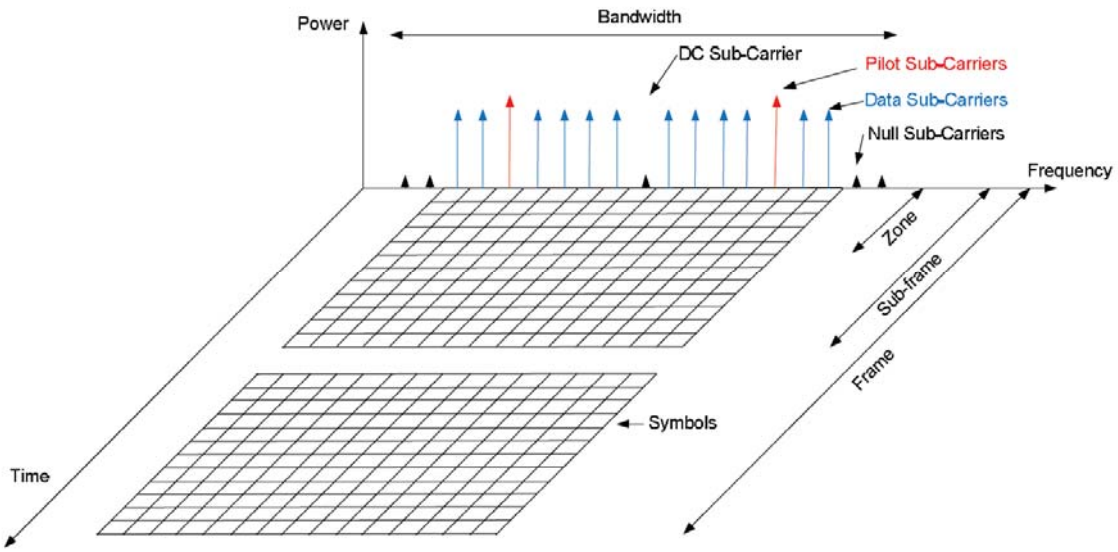

Figure 6. Complete OFDM signal illustrated in time, frequency, and power (From [39]).

These symbols are then grouped into subframes for transmission. In the IEEE 802.16d specification, there are two types of subframes as illustrated in Figure 7 [10]. The DL subframe begins with a preamble that is used for the physical layer procedures of time and frequency estimation and initial channel estimation. This is followed by the Frame Control Header (FCH) which provides frame configuration information and also the downlink map (DL-MAP) and uplink map (UL-MAP) message lengths. This is followed by the DL-MAP and UL-MAP sections that assign subsequent sections of the frame to each SS and define the modulation and coding schemes used in the link. The DL-MAP and UL-MAP sections are collectively referred to as the MAP. The MAP is followed by the DL subframe that contains data sent from the BS to the SS. The DL is divided up into time slices called "bursts." The different bursts are transmissions from the BS to separate SSs as assigned in the MAP. This is followed by a set of guard symbols where the channel is momentarily silent. The guard symbols signal the transition to the UL subframe that contains the data sent from the SSs to the BS. The UL is also divided up into time slices called bursts. The different bursts are transmissions from the individual SSs to the BS as assigned in the MAP. The UL section also contains

small sections that allows for ranging and for contention-based control signaling for the SS to establish a connection to the BS and for the SS to ask the BS for increased bandwidth. After the UL subframe, a set of guard symbols are transmitted again before the next frame's preamble begins again.

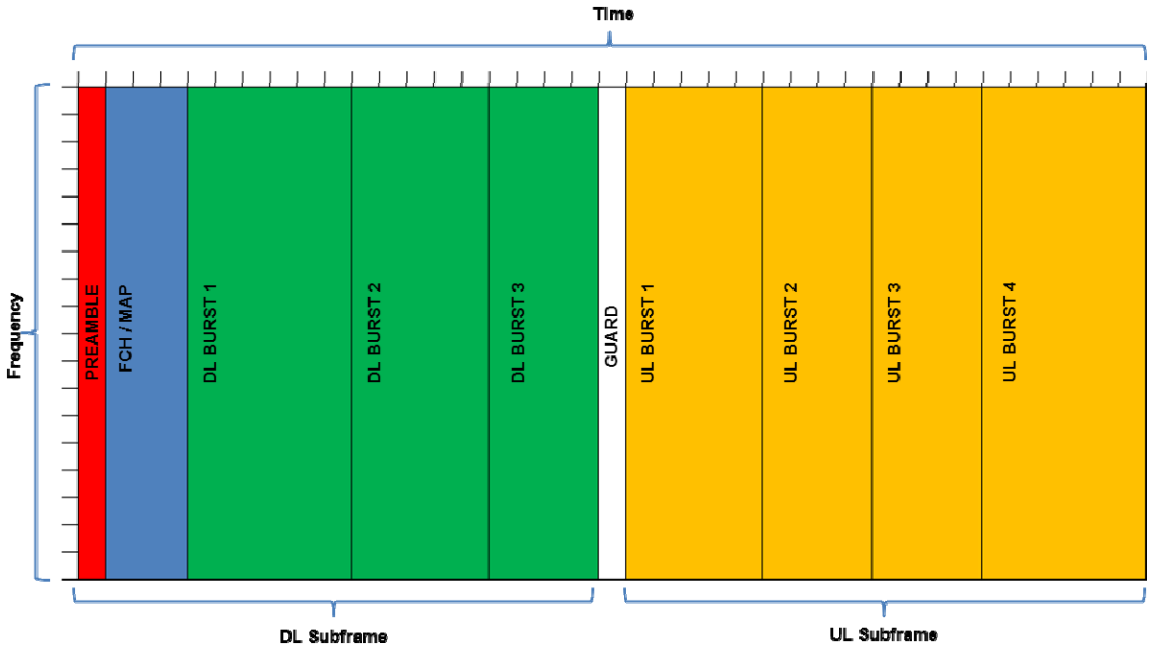

Figure 7. 802.16d downlink and uplink subframe division illustrated in frequency and time (After [40]).

The 802.16e standard breaks up the UL and DL subframes into smaller bursts that are not only allocated to the SSs according to time but also according to frequency as shown in Figure 8. This allocation scheme is called orthogonal frequency-division multiple access (OFDMA) and allows the network to make better use of the resources by dividing it up to more accurately reflect the network demands.

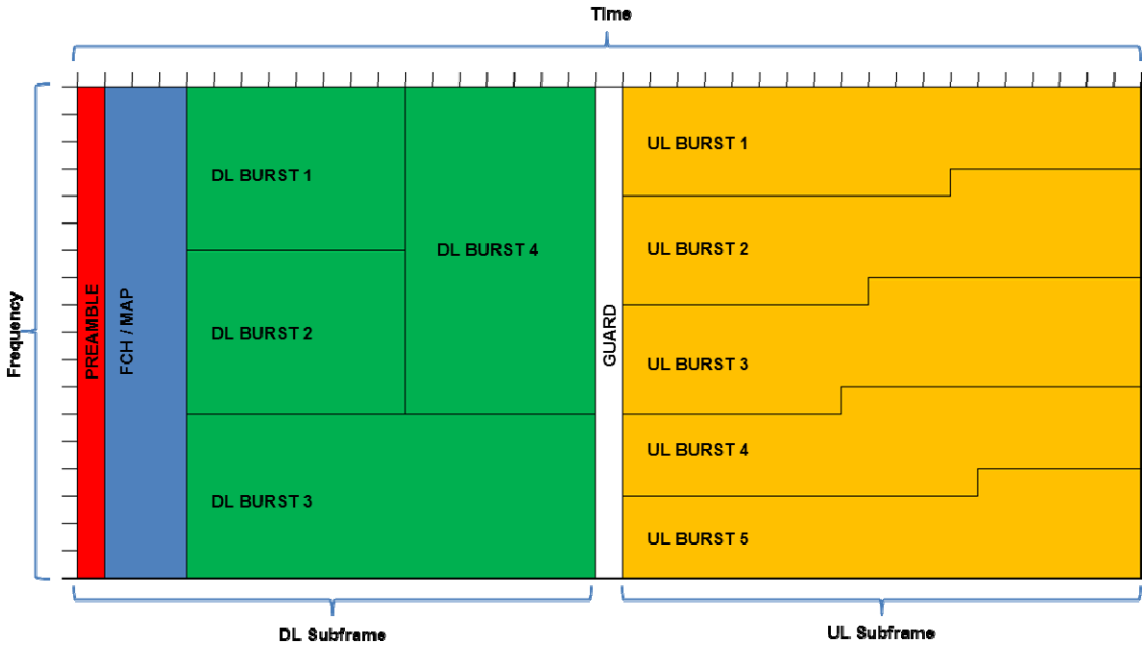

Figure 8. 802.16e downlink and uplink subframe division illustrated in frequency and time (After [40]).

B. MAC LAYER

The MAC layer receives data packets from the application layer (voice packets, data packets, etc.) called MAC service data units (MSDU) and organizes them into MAC protocol data units (MPDU) [10]. Sometimes it breaks up large MSDUs into smaller pieces to transmit them in parts and other times combines multiple MSDUs together to transmit inside of a MPDU as shown in Figure 9. Additionally, MPDUs often contain other types of network control information such as SS requests for more bandwidth or an automatic repeat request (ARQ) for the sender to repeat a message that was lost or corrupted. Handoffs of a SS from one BS to another and security functions are also handled in the MAC layer but are not covered in this thesis.

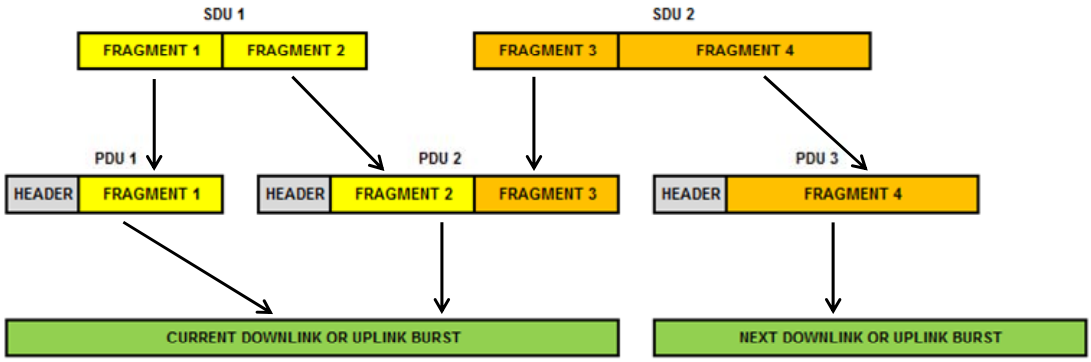

Figure 9. 802.16e segmentation and concatenation of SDUs in MAC PDUs (After [10]).

In order to simplify the modeled MAC layer, the separation of MSDUs into PDUs was minimized. SDUs were only fragmented in the model so that they could fit into the burst. Then each burst was given a header as shown in Figure 10. This was possible due to the large relative size of the implemented model bursts relative to the 802.16e standard bursts at the physical layer. Accordingly, the bursts are synonymous with PDUs in the implemented model.

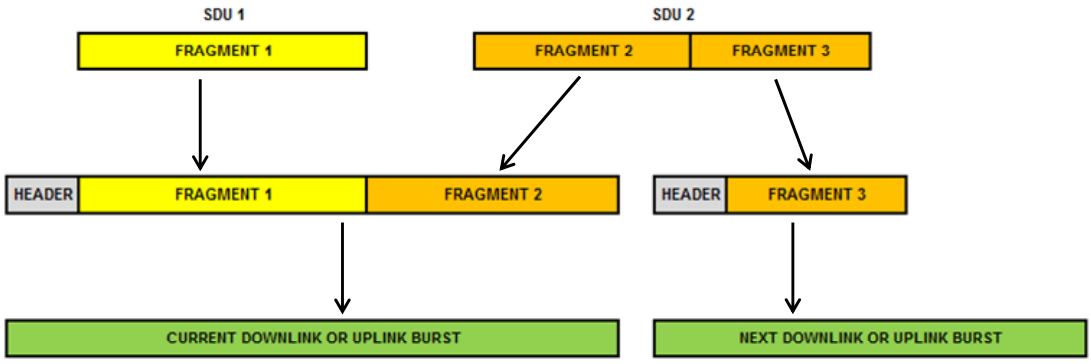

Figure 10. Model implemented segmentation and concatenation of SDUs (After [10]).

1. QoS Metrics

To provide QoS control of traffic sent over the network, 802.16 uses a connection-oriented MAC layer. Before a data transfer is commenced between the BS and the SS, a connection identifier (CID) is established between the two for both the UL and the DL connections [10]. This is a temporary address between the two network nodes which allows the MAC layer of the BS to talk directly the MAC layer of the SS. In the UV Sentry example, each vehicle would have two CID's, one for the DL connection and one for the UL connection. In addition to connections between the network nodes, service flows are also individually addressed with a service flow identifier (SFID). A service flow that was negotiated between the BS and the SS for a particular traffic flow (like video traffic) might include QoS parameters like those listed in Table 7. In the UV sentry example, the service flow for a vehicle's navigation control data is different than its FLIR video data. The 802.16 standard defines five scheduling services that characterize different traffic flow types.

Table 7. 802.16 service flow types, parameters, and equivalent UV Sentry traffic flows (From [10]).

Service Flow	**Description**	**QoS Parameters**	**UV Sentry Traffic**
Unsolicited grant services (UGS)	Supports fixed-size data packets at constant bit rates	Maximum sustained rate Maximum latency tolerance Jitter tolerance	Vehicle control data
Real-time polling service (rtPS)	Supports real-time services that generate variable-size data packets on a periodic basis	Minimum reserved rate Maximum sustained rate Maximum latency tolerance Traffic priority	Radar video
Nonreal-time polling service (nrtPS)	Supports delay tolerant data streams that require variable sized data grants at a minimum guaranteed rate	Minimum reserved rate Maximum sustained rate Traffic priority	
Best-effort service (BE)	Supports data streams that do not require minimum service-level guarantee	Maximum sustained rate Traffic priority	
Extended real-time polling service (ErtPS)	Supports real time applications that have variable data rates but require guaranteed data rate and delay	Minimum reserved rate Maximum sustained rate Maximum latency tolerance Jitter tolerance Traffic priority	FLIR video

A framework for scheduling traffic is provided in Table 7. However, the specific algorithms used by the 802.16 MAC layer to assign bandwidth for the UL and DL connections between the BS and the SS are undefined in the standard. Instead it is left to the BS development vendors to determine their own individual algorithms. Also, the SS development vendors are also free to choose their own algorithms to utilize the assigned bandwidth to best serve the above traffic flow types.

2. Relay Capability

The IEEE 802.16j amendment to the IEEE 802.16 standard allows the RS to relay messages received from other SS to the BS [35]. There are two cases where the use of a relay is more desirable than the use of a direct link between the BS and the SS. The first

case is where the SNR between the SS and the BS is so low that only low data rates can be supported. This can be caused either by a large distance between the BS and the SS or by a large amount of interference caused by sources external to the BS to SS link. In this case, a RS that is positioned between the BS and the SS may have a higher signal-to-noise ratio for both the BS and the SS links and be able to support higher data rates for both connections. However, this benefit comes with a tradeoff. If the OFDM radio uses a time-division scheme (like TDMA), then a radio without a relay will divide the time resources only between the links that directly connect the BS to the SS (BS-SS links). However, if relays are used, then a portion of the time resources will be allocated to both the BS-RS links and the links that directly connect the RS to the SS (RS-SS links). It is situation dependent whether the increase in SNR is worth this division of time resources. The second case where the use of a relay would be more desirable than the use of a direct link is when the SS is not within direct communication range of a BS. In this case, a RS or series of RSs can be placed between the BS and the SS in order to bridge the communications range gap.

Because of the interdependence between the physical and MAC layers in the 802.16 standard, relay cannot be accomplished effectively at only the physical level. Thus, a simple repeater RS solution that "listens" for the SS transmission and repeats the transmission verbatim to the BS (and vice versa) is not an implementable solution. This is due in part to the centralized and scheduled MAC protocol implemented in the 802.16 standard. The BS allocates channel resources via the MAP for each burst transmission. The message bits from each SS that are transmitted in these bursts form a cohesive unit with a header in the beginning of the burst that contains essential information about the burst (burst origination, burst type, burst length, etc.) that are translated into OFDM symbols before transmission. Since the number of bits that can be allocated per symbol is SNR dependent and the scheduled allocation for the BS-RS link is not guaranteed to be as large as the for the RS-SS link, there necessarily must be buffering that occurs at the RS. In order to ensure that the messages are buffered at the RS in a manner that they can be retransmitted to the BS, they must be decomposed from the bursts into protocol data units (PDU) that can be buffered and then reassembled onto different sized bursts for a

subsequent transmission to the BS. This decomposing, buffering, and reassembling must be done at the MAC level and adds an additional level of complexity to the RS.

Because the RS operates at the MAC layer, it also has the ability to add and remove traffic from the network. Accordingly, a RS can have individual unique CIDs and SFIDs like a SS to support service flows. In the UV Sentry example, a vehicle can be a RS that is relaying traffic from other vehicles while also transmitting and receiving network traffic that originates and terminates onboard that vehicle. This means that a UV Sentry vehicle can receive and transmit vehicle control messages while also relaying a FLIR video that originates on another vehicle. Thus, the relaying vehicles function as more than simple relays; they can also accomplish other functions for the UV Sentry network while also relaying messages.

a. Standard Specifications

Depending on the purpose of the relay, we find that the 802.16j standard offers two ways to implement the relay—transparent and nontransparent relays. The transparent relay is for the first relay case discussed where all the RSs and SSs are within range of the BS, and a higher data rate is desired in the coverage area. The nontransparent relay is for use in the second relay case discussed, where not all SSs are directly within range of the BS. In this case, RSs are necessary to extend the range of the communications network.

The transparent relay's frame implementation is illustrated in Figure 11. The BS begins the frame by transmitting a preamble and then allocating channel resources directly by transmitting the MAP to both the RSs and the SSs. This transmission occurs at a very low order modulation rate like binary phase-shift keying (BPSK) to ensure that it can be received reliably by even the distant SSs and RSs [10]. After the MAP is transmitted, message data is transmitted at higher order modulation rates. In the specification, the DL subframe and the UL subframe are each divided for sharing between the BS and the RSs. In the first part of the DL subframe after the BS has transmitted the MAP, the BS transmits data to the RSs (for further relay to SSs). In the second part of the DL subframe, the medium is shared as specified in the MAP between

DL transmissions from the BS to the SSs it directly serves and also DL transmissions from the RSs to the SSs it directly serves. In the UL subframe, a similar access scheme is used. In the first part of the UL subframe, the medium is shared between UL transmissions between the RSs and the SSs it directly serves and also UL transmissions between the BS and the SSs it directly serves. In the second part of the UL, the BS receives the relayed transmissions from the RSs the BS directly serves. The process then repeats indefinitely.

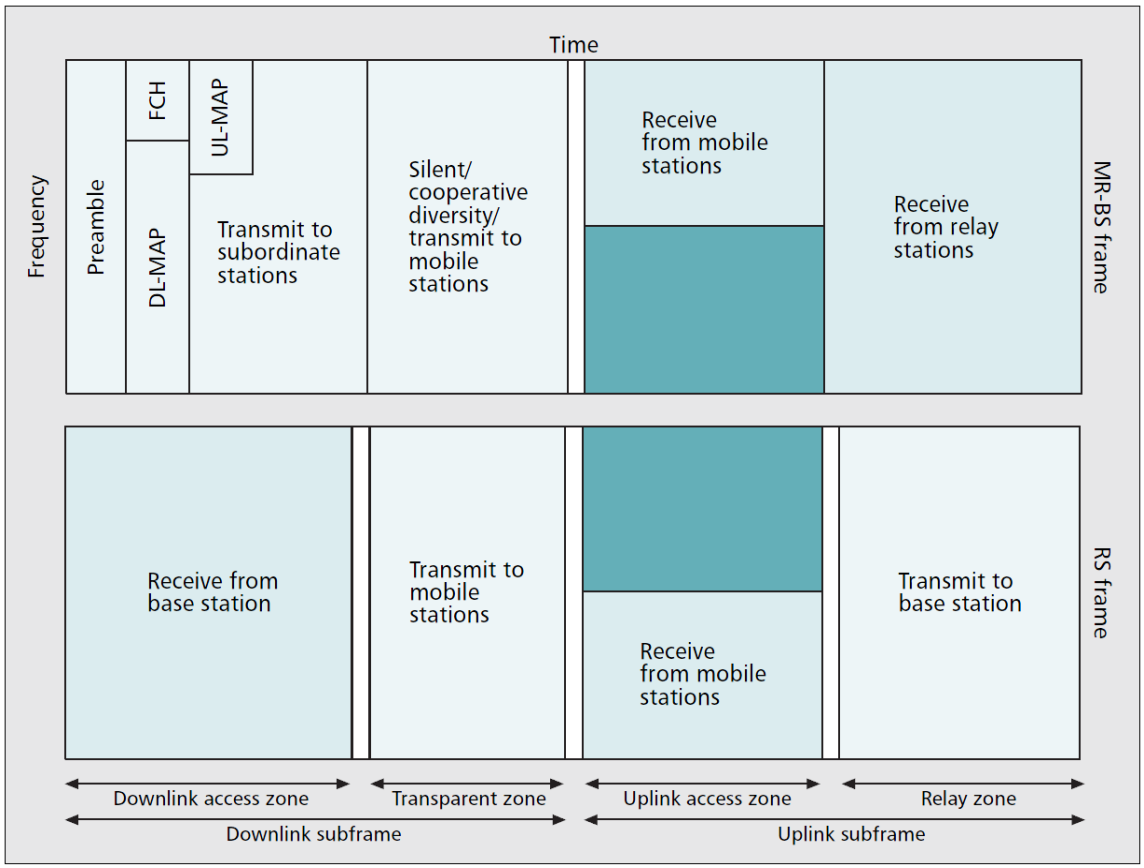

Figure 11. Transparent relay frame where the upper block depicts the actions of the BS and the lower block depicts the actions of the RS (From [26]).

The nontransparent implementation is illustrated by Figure 12. The BS and the RSs simultaneously transmit a preamble and then allocate channel resources by simultaneously transmitting their MAPs at a low modulation rate. The MAP transmitted by the BS is received by the SS it directly serves and also by the RSs it directly serves.

The MAP transmitted by the RSs is received by the SSs that they directly serve. After the MAP is transmitted, message data is transmitted at higher modulation rates. In the first part of the DL subframe, the BS transmits data to the SSs it directly serves, and the RSs also simultaneously transmit data to the SSs that it directly serves (using the same frequencies). In the second part of the DL subframe, the BS transmits data to the RSs that it directly serves. In the first part of the UL subframe, the BS receives data from the SSs it serves, and the RSs simultaneously receive data from the SSs that they serve (using the same frequencies). In the second part of the UL subframe, the BS receives the transmissions from the RSs it serves. The process then repeats indefinitely.

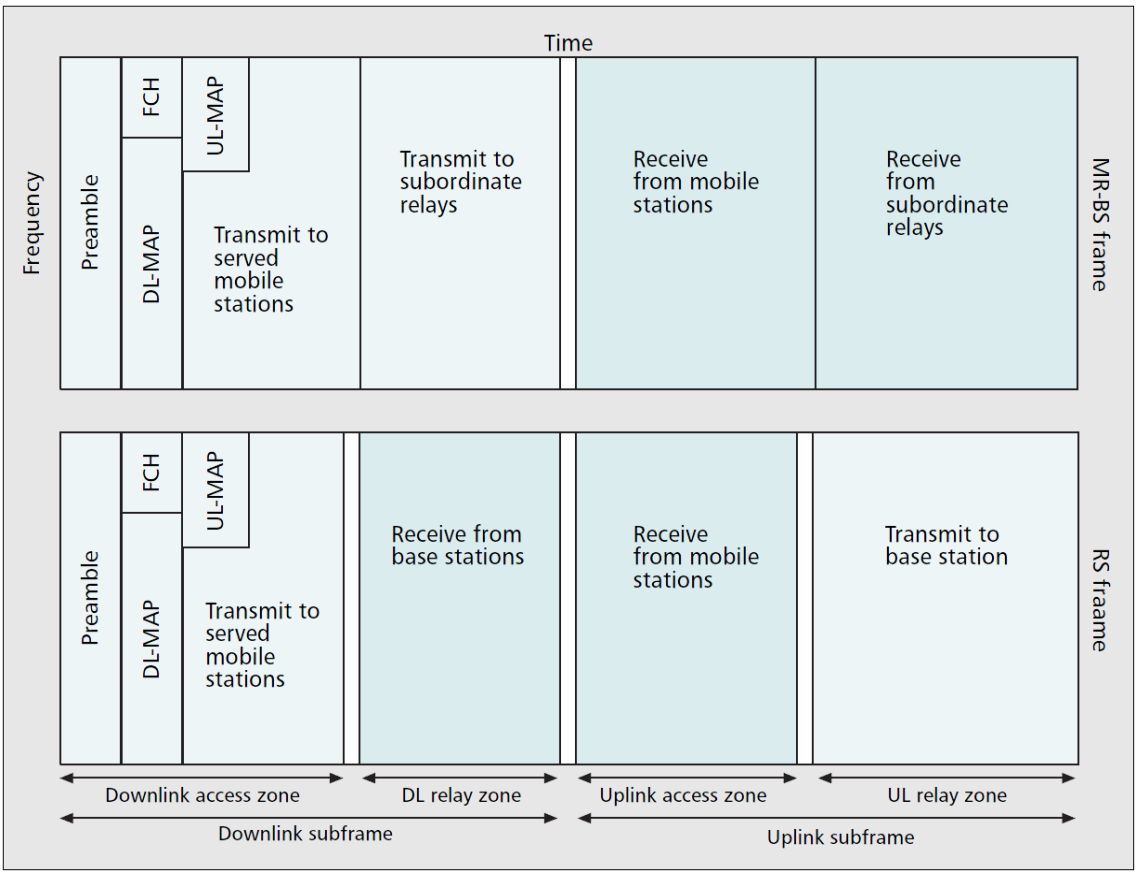

Figure 12. Nontransparent relay frame where the upper block depicts the actions of the BS and the lower block depicts the actions of the RS (From [26]).

Because the nontransparent relay implementation requires simultaneous transmissions in the same frequency, there is a possibility for increased interference in the

wireless communications network as discussed in the exposed node and hidden node problems above. This is especially important for the preamble and MAP; however, it also becomes a significant factor for the data transmissions when the relayed SSs are too close to the nonrelayed SSs [26]. While power control can be used to keep the RSs and SSs from substantially interfering each other, the nontransparent relay implementation is best used in the case for where it was designed – to connect SSs to the network that are beyond the direct communications range of the BS.

One of the benefits of using nontransparent relays is that they can operate in "distributed scheduling mode, where they make decisions about resource allocation to their subordinate stations, possibly in coordination with the [BS]." [26] In the traditional 802.16 network, the BS makes all the resource allocation decisions based on the requests that it has received from the SSs. However, because the RSs operate at the MAC level in order to read the burst headers in order to buffer and route traffic, they are also privy to reading the SS allocation requests and can be knowledgeable about the individual connections and negotiated service flows. Equipped with this information, a RS that coordinates with the BS can make the same types of resource allocation decisions that the BS makes. In a distributed scheduling mode, the RS assumes some of this responsibility. In this mode, the RS reads the service flow requests that come from the SS it serves and effectively sums them up into a single request. This request is then relayed to the BS. The BS then allocates to the RS a block of channel resources that are sufficient to meet the RS's cumulative service flow needs, and the RS subdivides this block of resources among its SSs based on the individual negotiated service flows. In a sense, the RS acts like a BS for all the SSs that it services.

b. Implementation

The transparent and nontransparent implementations of 802.16j relay discussed above each operate under specific assumptions. In the transparent relay, it is assumed that all SSs and RSs are within direct communication range of the BS. In the modeled scenario, this assumption can assumed to be true based on the physical layer assumptions listed below and the line-of-sight (LOS) equation [28]

$$d = 3.57\left(\sqrt{Kh_1} + \sqrt{Kh_2}\right) \quad (1)$$

where d is the maximum distance between two antennas for LOS propagation in km, h_1 is height of antenna one in m, h_2 is height of antenna two in m, and K = 4/3. However, the UV Sentry Design Reference Mission lists other missions that UV Sentry could be tasked to complete where the patrolled geographical area is much larger. In this case, the assumption that the BS is always within direct communication range of the SS cannot be assumed to be true. In fact, it is this range extending capability of the 802.16j relay system that makes it desirable for use in UV Sentry communications system. Accordingly, the transparent implementation of 802.16j is not considered for the UV Sentry system.

The nontransparent implementation of the 802.16j relays assumes that the SS connected to the network through a RS are sufficiently far from the rest of the network nodes that there is not a substantial amount of interference between the simultaneous BS-SS transmissions and the RS-SS transmissions. However, this is not the case for the modeled scenario where all network nodes are confined to a small geographical area. In that case, the amount of mutual interference will likely be substantial. Accordingly, the nontransparent and transparent relay implementations are mutually exclusive in the types of cases in which they should be deployed. The nontransparent relay increases the available data rate in a small coverage area, and the transparent relay expands the coverage area beyond LOS distances. Neither relay implementation seems flexible enough to accommodate the UV Sentry communications network changing topology and multiple missions. As the UV Sentry vehicles conduct their patrols, they may constantly be moving into and out of LOS range of a BS. Accordingly, a modified implementation of the nontransparent 802.16j implementation was created to meet the requirements that the relay network operate in both geographically confined and distant scenarios.

The main reason why the 802.16j nontransparent relay is not an acceptable solution for the UV Sentry communications network is because it allows simultaneous transmissions by separate nodes using the same frequencies that have the potential of

causing substantial interference. To resolve this problem, the possibility of more than one vehicle simultaneously transmitting in the same frequency was eliminated. The 802.16 nontransparent relay frame shown above split the DL and UL subframes into two parts that were shared by the BS and the RSs. To implement this scheme without simultaneous transmissions would require significant coordination between the BS and the RS so that the RS could transmit in the subframes as soon as the BS was finished (and vice versa). Instead of splitting the UL and DL subframes between the RS and BS, a separate transmission frame was allocated for the RS-SS link transmissions, and the BS and RS alternate in transmitting their frames. The control of the medium alternates between the BS and the RS as shown in Figure 13. In the BS controlled transmission, the BS transmits the preamble, MAP, and DL data just like in the nonrelay case, but the recipients of the data include RSs as well as SS. The BS then receives data in the UL from SSs and RSs. The next frame is then controlled by the RS. The RS transmits the preamble, MAP, and DL data to the SSs it serves in a similar manner as the BS did in the previous frame. The RS then receives data in the UL from the SSs it serves. The process then repeats again indefinitely beginning with the BS controlled frame.

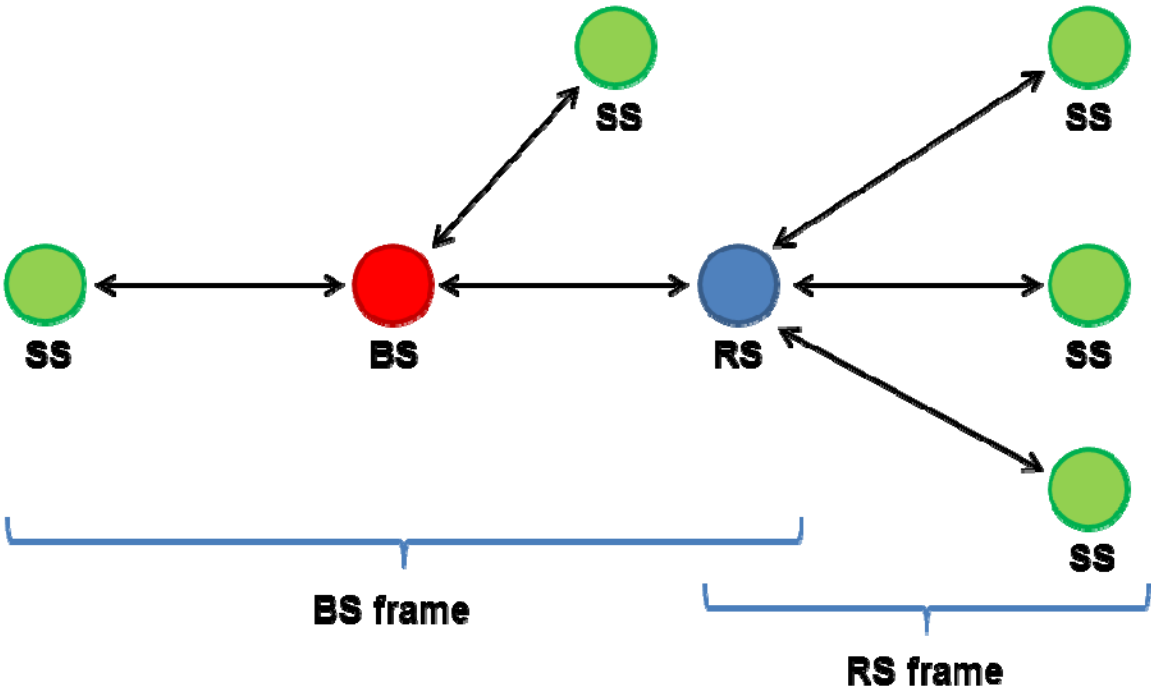

Figure 13. Example two-hop relay topology and frame sequence.

This scheme makes use of the distributed scheduling mode discussed above. The BS does not individually allocate channel resources to the SSs served by the RS. Instead, this responsibility is left to the RS. The BS allocated resources to the individual RSs in large blocks according to the cumulative service flow requests transmitted by the RSs. This is transmitted to the RSs in the BS frames MAP. The RSs upon receiving the MAP subdivide their allocated portion of the RS frame amongst their own needs and SSs that they serve. In essence, this relay system becomes a form of tunneling where the intermediate RSs need not be aware of the individual connections and simplifies the resource allocation task [26].

An example of how the BS may allocate the RS frame is shown in Figure 14. For the model implementation, the RS frame allocations were accomplished through the allocation of a large block of subcarriers collectively called a channel. In the example, the BS allocated the first three channels of the RS frame for the use of one of the RSs it was directly connected to (RS1A) and allocated the fourth channel for the use of another RS that it was directly connected to (RS1B). Accordingly, RS1A and RS1B use these channels in the RS frame to talk to their subservient SSs. As these are only allocations, RSs decide who among their subservient SSs or subservient RSs will be given access to the network by transmitting that information in the RS frame MAP.

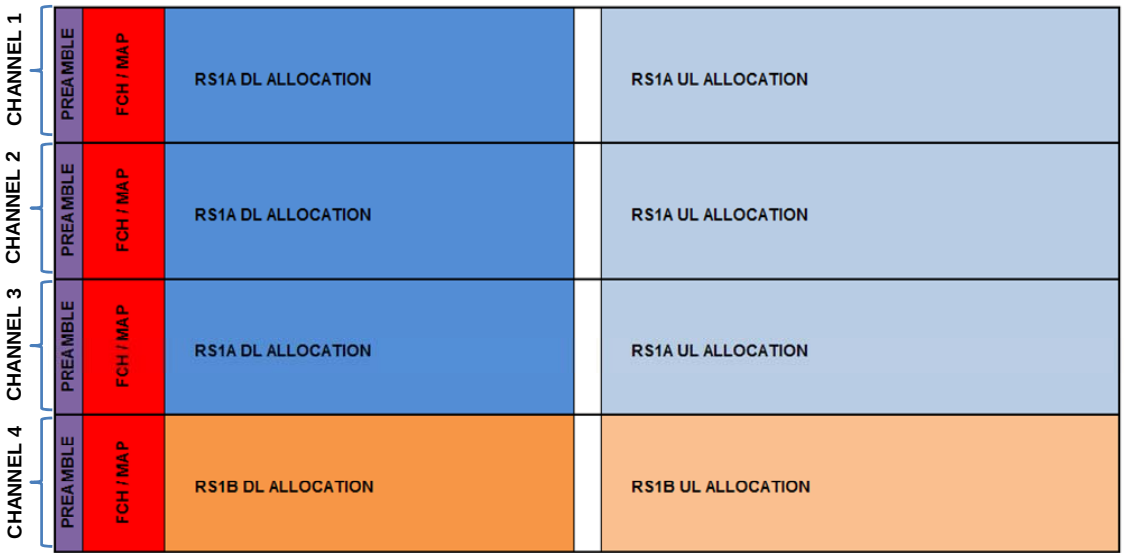

Figure 14. Example RS frame allocations in frequency via four simultaneous channels.

This scheme also allows for allocation flexibility. In the event that the RSs do not require the complete RS frames resources, the BS could possibly only allocate some of the RS frame channels to the RSs. The BS could then use the other channels of the RS frame for transmissions between the BS and the SSs. An example of this is shown in Figure 15. In this case, the BS kept channel 1 of the RS frame for its own use, allocated the second and third channels for the use of one of the RSs that the BS was directly connected to (RS1A), and the BS allocated the fourth channel for the use of another RS that it was directly connected to (RS1B). Accordingly, RS1A and RS1B use these channels to talk to their subservient SSs during the RS frame. As these are only allocations, RSs decide who among their subservient SSs or subservient RSs are given access to the network by transmitting that information in the RS frame MAP. However, this case was not implemented in the modeled relay. Instead, the entire RS frame was dedicated to transmissions between RSs and SS.

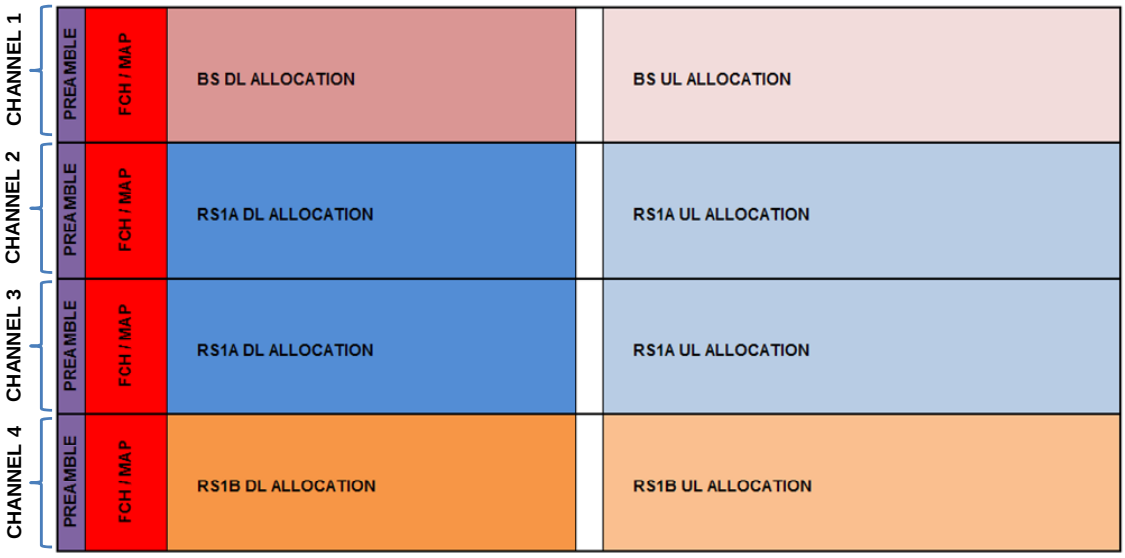

Figure 15. Example RS frame allocations in frequency via four simultaneous channels. In this example, the BS keeps a single channel for its own use and allows the RS to make use of the other three for communication with their subservient SSs.

This scheme is also not limited to a single relay, but because of distributed scheduling, it can be scaled so that multiple relays may be implemented. For a three hop relay system as shown in Figure 16, instead of being a single RS frame there could be a relay station 1 (RS1) frame and a relay station 2 (RS2) frame. The frame transmission order would start with the BS frame. The BS frame is then followed by the RS1 frame, and that frame is in turn followed by the RS2 frame. The process then begins again with the BS frame. The scheduling and allocation also works as just described. The furthest relay from the BS (RS2) transmits the cumulative resource request from the SSs it is servicing (and its own resource requests) to RS1. RS1 then incorporates these resource requests with its own resource requests and the requests from the SS it is servicing when it sends its resource request to the BS. The BS then allocates the next two RS frames to RS1 via the BS frame MAP. RS1 then allocates resources to RS2 (and the SSs RS1 may be serving) via the RS1 frame MAP. RS2 then finally allocates resources to the SS it is serving. The tradeoff with the above multi-relay implementation is that as the number of relays increases, the amount of overhead and time delay in transmissions also increases. The multi-relay implementation was not modeled.

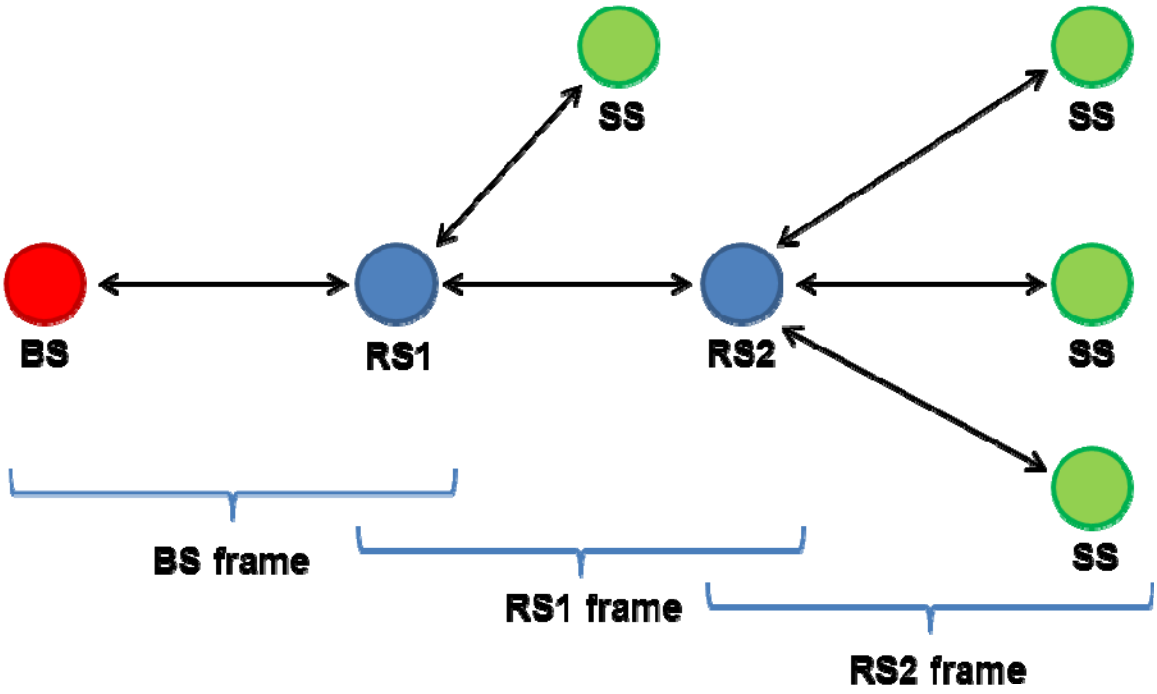

Figure 16. Example three-hop relay topology and frame sequence.

As discussed in Chapter II, the communications system must be robust in order to allow coordination between vehicles, even if communication back to the central node is lost. In order to achieve this end, a form of clustering is used. Using the vehicles in the above relay as an example, if all of the above vehicles were to maneuver out of communications range of the central BS, they should still be able to cooperatively operate to achieve the mission. This is because, in essence, each RS in Figure 16 is acting like a BS. Accordingly, if each vehicle is equipped with a radio that is capable of acting like a RS if necessary and like a SS otherwise, then the resulting network is highly resilient. When the network branch above maneuvers away from the BS, an administrative message can be broadcast throughout the branch to elect a new BS. Due to the distributed COP that each vehicle helps to maintain, the network topography is known. The vehicle that occupies the most central location can be elected as the BS radio. For

this example, RS2 will be elected as the new BS, and the resulting network will now be a single relay network, as depicted in Figure 17. This feature was not included in the implemented model.

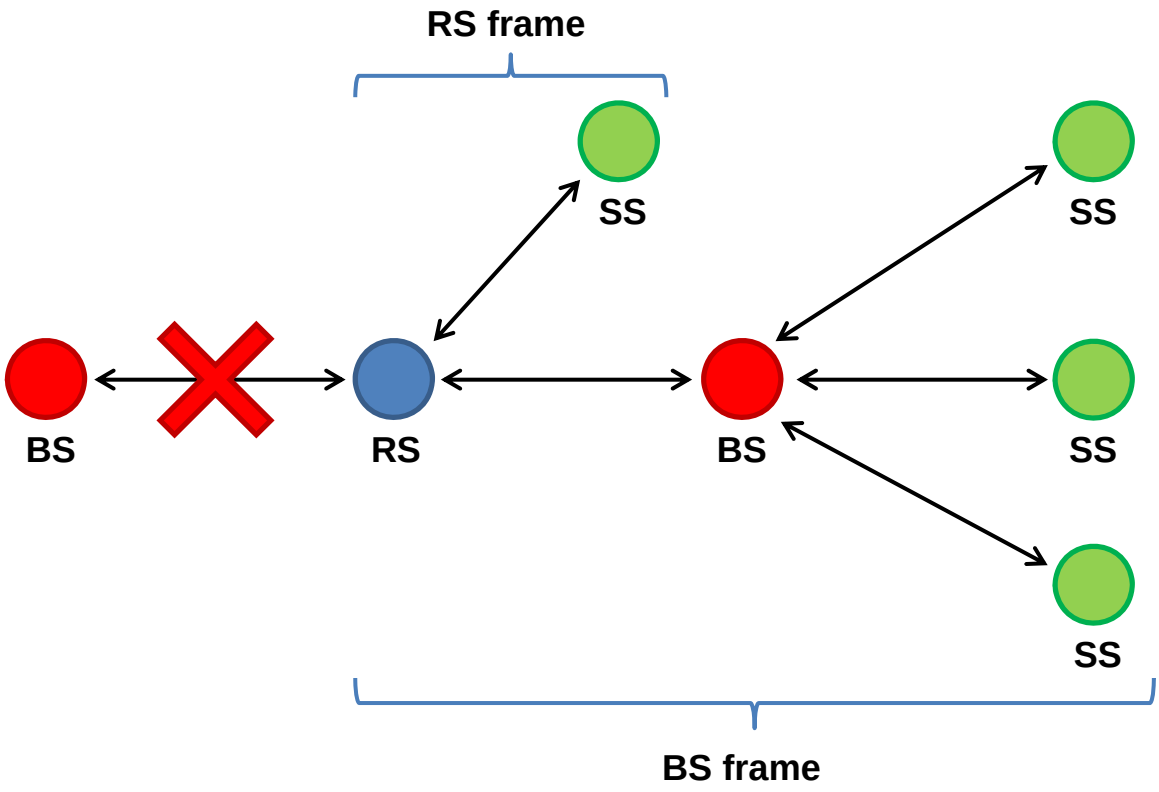

Figure 17. Example of RS assuming BS duties when disconnected from original BS.

V. UV SENTRY NETWORK MODEL AND SIMULATION

There were a number of networks simulators available to simulate the UV Sentry network. The most notable of these the OPNET Modeler simulator and the QualNet Developer simulator offered by Scalable Network Technologies. The OPNET Modeler offers a Wireless Suite that allows for the simulation of either the fixed IEEE 802.16-2004, the mobile 802.16e, or the transparent 802.16j networks [41]. Similarly, QualNet Developer offers an Advanced Wireless Library that allows for the simulation of either the fixed IEEE 802.16-2004, the mobile 802.16e, or the transparent 802.16j networks [42]. If we had determined that one of these 802.16 network types would be a sufficient network protocol for UV Sentry, then these networks simulators would have been ideal for simulating the UV Sentry communications network.

As discussed above, the 802.16 standard was necessarily modified in order to best suit UV Sentry. The source code for these network simulation models is provided with the software, and in theory the MAC layer could have been modified in order to simulate the desired UV Sentry relay model. However, in practice we found that the effective modification of these models source code was beyond our ability. Thus, a suitable UV Sentry MAC layer and application layer was written in MATLAB code as embedded function Simulink blocks. These block models interfaced with an 802.16d physical layer model that was included in the 2011a release of MATLAB [43]. This original Simulink physical layer model was significantly modified to interface with the created MAC layer blocks. In so doing we created a MAC layer to precisely implement the relaying MAC layer discussed above, and then we compared its performance to that of a similar but nonrelaying MAC layer. A diagram of the complete Simulink model and the accompanying embedded function MATLAB source code is included in Appendix A. A list of modeled functions is included in Appendix B. Additionally, the reasons for the pertinent model parameters, constraints and abstractions are discussed.

A. MODELED APPLICATION LAYER

The application layers for the BS and the SS primarily create packets to be transmitted to other nodes and accept packets that were transmitted to them. The traffic was modeled to be as realistic as possible based on the discussions in Chapter II. Once a packet is created by an application layer, it is placed in its corresponding queue and waits for the MAC layer to access the queue. At the other end, the application layer accepts packets delivered to it by the MAC layer and then reads the data inside the packet in order to accomplish such tasks as controlling the aircraft, recreating a radar video image, or recreating a FLIR video image.

1. BS Application Layer Functions

DL vehicle control packets originate at the BS application layer and are delivered to the BS MAC layer for transmission to the SS for ultimate delivery to the SS application layer. In manual control, the transmitted DL vehicle control messages direct the vehicle movement, radar settings, and FLIR settings. In automatic control, the transmitted DL vehicle control messages facilitate vehicle coordination. The data contained in these messages are further described in Table 8. Manual vehicle control for an UAV can be accomplished by a 12800 bps data stream [44]. The method in [44] is a very direct method of vehicle control that requires continuous instruction for the manipulation of the vehicle control surfaces. Even in manual mode, the UV Sentry system is not envisioned to require this high level of human control. Instead, the operator may input waypoints, turn, or speed commands for the vehicle. Also, the operator may manually change the settings or manually control the radar or FLIR. In automatic mode, the UV Sentry system will be able to coordinate its own movements and not require any BS operator input. Given this automated nature of the UV Sentry system, we believe that the constant data rate of 12800 bps per vehicle is a conservatively high estimate for the DL vehicle control message traffic profile. To simulate this data stream, transmissions were sent in 512 bit packets, were generated by the BS application layer every 35 ms for each vehicle, and were then passed on the BS MAC layer for transmission.

Table 8. DL vehicle control message data.

Message Type	**Data**
Manual Vehicle Control	Waypoint Turn Command Speed
Manual Radar Control	Scan Rate Range
Manual FLIR Control	Orientation Zoom Focus Automatic Target Lock
Automatic Vehicle Control	Common Operating Picture Vehicle Coordination Current Tasking

2. SS Application Layer Functions

UL vehicle control packets, radar packets, and FLIR video packets originate at the SS application layer and are delivered to the SS MAC layer for transmission to the BS for ultimate delivery to the BS application layer. The transmitted UL vehicle control messages contain information concerning the equipment status, equipment faults, fuel status, navigation and orientation information. The data contained in these messages is further described in Table 9. It has been shown that a constant data rate of 12800 bps is a conservatively high estimate for these UL vehicle control messages [44]. Again, the transmissions were sent in 512 bit packets and were generated by each SS application layer every 35 ms in order to be passed on the SS MAC layer for transmission.

Table 9. UL Vehicle control message data.

Message Type	**Data**
Equipment Status	Engine Torque Engine Temperature Engine Pressure Transmission Temperature Transmission Pressure
Equipment Faults	Error Code
Fuel Status	Fuel Level
Navigation Information	Position Heading Speed Elevation
Orientation Information	Pitch Roll Yaw

FLIR and radar video packets were modeled as variable bitrate transmissions. The typical FLIR video capable of being carried onboard a VTUAV or USV is characterized by a 640 x 480 pixel resolution with 8-bit pixels that are updated 15 frames per second (fps) [17], [45], [46]. To model the radar data traffic, a typical commercial surface search radar was used. This radar produces a 480 x 480 pixel resolution with 8-bit pixels that were updated every 0.4 seconds [47]. Video data has a high amount of redundancy that can be reduced through compression techniques. Levels of compression for FLIR video has been shown to be 46:1 using traditional MPEG techniques, and specialized algorithms can deliver performance up to 256:1 [48]. Compression levels for radar were similarly shown to be 50:1 [49]. Thus, the average packet size that updates the radar or video image $E(x)$ is substantially reduced.

MPEG2 compressed video traffic packets have been shown to be widely variable in size and exhibit self-similar characteristics [50]. The Hurst parameter is used to measure the amount of self-similarity or "burstiness" that causes the packet size to vary widely [51]. The video packet sizes in [50] were self-similar with a Hurst parameter that

ranged between 0.824 and 0.979. The average of the Hurst parameter values (H = 0.9055) given in [50] was used for modeling video packet size variability. To model the packet size distribution, a Pareto distribution was used because of its high kurtosis [51]. The Hurst parameter was converted to the Pareto distribution parameter α using [51]

$$H = \frac{(3-\alpha)}{2} \quad . \tag{2}$$

The probability density function for the Pareto distribution is given by

$$f(x) = \frac{\alpha b^{\alpha}}{x^{\alpha+1}} \quad , \quad x \geq b \tag{3}$$

where b is the minimum packet size in bits. Using $E(x)$, α, and the formula for the Pareto distribution expected value

$$E(x) = \frac{\alpha b}{\alpha - 1} \quad , \quad a > 1 \tag{4}$$

we can calculate b. Using the calculated values of α and b given in Table 10 we can vary the length of each FLIR and radar image update packet in the simulation to follow the Pareto distribution.

Table 10. Variable video traffic parameters for UV Sentry FLIR and radar traffic.

Traffic Type	**fps**	**x pixels**	**y pixels**	**bits / pixel**	**comp. ratio**	$E(x)$ **(bits)**	H [50]	α	b **(bits)**
FLIR	15	640	480	8	46:1	53426	0.9055	1.189	8576
Radar	0.4	480	480	8	50:1	36864	0.9055	1.189	5920

B. MODELED MAC LAYER

The primary functions of the modeled BS MAC are to convert MAC bursts into application packets, to convert application packets into MAC bursts, to control the flow between the two layers, and to allocate resources to the SS. Similarly, the primary functions of the modeled SS MAC are to convert MAC bursts into application packets, to convert application packets into MAC bursts, and to control the flow between the two layers. While the BS and the SS share the same overlying functions due to the BS-SS hierarchy and a delay in network status updates, the BS and SS differ in the way that they

execute these functions. It is important to note that due to the way the relay is implemented, vehicles that occupy a relay function receive relay packets as if they are a BS and then transmit the relay packets to the actual BS as a SS.

1. BS MAC Layer Functions

In order for the BS to transmit received application packets in the UL, they must be converted into MAC transmission bursts. The highest priority packet is taken from the highest priority service flow until that queue is empty, and then the next highest queue is accessed. The priority of the SFID flows is in the following order: control data, FLIR video data, and then radar data. However, the BS does not transmit FLIR or radar packets in the UL channel to the SS, so the control data is the only data transmitted. Also, network status packets are not transmitted to the SSs because the SSs are not required to know the size of the BS SFID queues. A packet header is appended to the beginning of each packet received from the application layer. This packet contains the CID, SFID, packet length, sequence number, and time that the packet was first created. The MAC burst is assembled from these packets, and if the burst becomes full, the last application packet is broken up into two subpackets. The first subpacket part is sent in the current burst, and the remainder is sent in the next burst. A 48-bit header is appended to the beginning of each burst that includes the CID and burst length. The burst is then sent to the BS physical layer so that it can be transmitted.

To convert MAC bursts into application packets, the BS MAC layer first receives the UL transmission from the physical layer for each channel. Since the BS sent the MAP, the BS knows what resources are allocated to which SS and expects the SS to conform. Accordingly, the BS already knows the unique CID, the modulation rate, and the place in the UL transmission where the CID transmission burst begins and ends. Once the BS finds a burst, the BS MAC layer reads the 48-bit burst header to confirm the CID, burst length, burst order and to accomplish a burst header check by using a header check sum. If the burst header is corrupted, the entire burst is discarded. Otherwise, the burst header is then discarded, and the burst is then combined in a predetermined way with other bursts that were transmitted in other channels and the partial packets that were

transmitted in the previous burst to form a superpacket. The superpacket is then broken up into application packets by reading the application packet length and removing that length from the superpacket. The application header also contains information on to which application SFID it should be routed to, the sequence number of the packet, the time it was sent, and the CID it was meant for. If the CID shows that the packet should be relayed to a subsequent station, it is then queued for retransmission. Also, due to QoS requirements, NRT FLIR video requirements are useless if they have been received later than 0.3 seconds after they were created. Accordingly, these packets are discarded at the destination's MAC layer if they are late. Otherwise, the packet is sent to the application level.

The BS MAC control receives packets from its application layer that are meant to be transmitted and receives packets from other SSs that are meant for relay. It places the packets in the appropriate SFID queue for that application packet type. All queues operate by a first-in first-out principle. The queues were created to be large enough that they would not overflow. The packet's length, time, CID, and SFID are also recorded. If the packet is a relay packet, the original packet information is recorded and not updated when placed in the new queue.

The BS MAC also has the responsibility to allocate resources for the next transmission via the MAP section of the DL transmission. To allocate the DL bursts, it evaluates the SFID queues for each CID by reading the SFID type and SFID queue size (total bytes). Using the SFID queue sizes as a guide, we see that the MAC allocation scheme uses a round robin algorithm to allocate a minimum amount of bandwidth for UGS applications and a weighted fair queuing algorithm to give priority to the vehicles with the highest SFID priorities until the BS runs out of resources. If in relay mode, the relay cycle plays a factor in allocation. If the cycle is allocated to the BS-RS link, then only vehicles directly connected to the BS are allocated resources. If the cycle is allocated to the RS-SS link, then it is the RS who allocate the resources to its subservient SS.

Allocation of the UL bursts is accomplished in a similar manner. Instead of being able to read the SFID queues directly, the BS is dependent on the status messages it

receives from the SS MAC layers to update it on the size of the SS SFID queues. If the BS did not receive a status message during the last transmission from a SS, it assumes that the status message was corrupted and increases the UGS service in order to clear the transmission of that CID's current application packet and receive a status message. Using the SFID queue sizes as a guide, the BS MAC allocation scheme uses a round robin algorithm to allocate a minimum amount of bandwidth for UGS applications and a weighted fair queuing algorithm to give priority to the vehicles with the highest SFID priorities until the BS runs out of bandwidth. If in relay mode, the relay cycle plays a factor in allocation. If the cycle is BS-RS, then only vehicles in that link may be allocated resources. If the cycle is RS-SS, then only vehicles in that link may be allocated resources.

2. SS MAC Layer Functions

The SS MAC control receives packets from the application layer that are meant to be transmitted and receives packets from other BS that are meant for relay. It then places the packet in the appropriate SFID queue for that application packet type. All queues operate in a first-in first-out principle. The FLIR video queue is managed to control SFID latency because it is a NRT service. If the latency of the oldest packet in the queue is greater than 0.25 seconds old, the queue is cleared and the new video application packet becomes the first packet written to the queue. The packet's length, time, CID, and SFID are also recorded. If the packet is a relay packet, the original packet information is recorded and is not updated when placed in the new queue.

In order for the SS to transmit application packets, the packets must be converted into MAC transmission bursts. The highest priority packet is taken from the highest priority service flow until that queue is empty, and then the next highest queue is accessed. The priority of the SFID flows is in the following order: control data, FLIR video data, and then radar data. Also, along with the application packets, a small network status packet is included in each transmission. The network status packet contains the SFID queue length (number of messages) and SFID queue size (total bytes) for each SFID queue type. This is a MAC layer message sent from the SS MAC to alert the BS

MAC of how much data of what type is waiting to be transmitted – allowing the BS to adjust its allocation in the next MAP. A packet header is appended to the beginning of each packet. This packet contains the CID, SFID, packet length, sequence number, and time that the packet was first created. The MAC burst is assembled from these packets, and if the burst becomes full, the last application packet is broken up into two subpackets. The first subpacket part is sent in the current burst, and the remainder is sent in the next burst. A 48-bit header is appended to the beginning of the burst that includes the CID and burst length. The burst is then sent to the physical layer so that it can be transmitted.

To convert MAC bursts into application packets, the MAC layer first receives the DL transmission from the physical layer for each channel. This transmission contains the MAP and the DL bursts for that channel. By first reading the MAP, the MAC can then locate the bursts sent by the BS to that specific SS. The MAP contains information that identifies the vehicles unique CID, the DL modulation rate, and the place in the DL transmission where the CID's transmission burst begins and ends. The MAP also tells the SS of the UL modulation rate, and beginning and ending of the allocation for the SS UL CID burst. Using the MAP, the SS can locate the DL burst(s) that were sent to it by the BS. Once it finds a burst, it reads the 48-bit burst header to confirm the CID, burst length, burst order and to accomplish a burst header check by using a header check sum. If the burst header is corrupted, the entire burst is discarded. Otherwise, the burst header is then discarded, and the burst is then combined in a predetermined way with other bursts that were transmitted in other channels and partial packets that were transmitted in the last burst to form a superpacket. The superpacket is then broken up into application packets by reading the application packet length and removing that length from the superpacket. The application header also contains information on which application SFID the packet should be routed to, the sequence number of the packet, the time it was sent, and the CID it was meant for. If the CID shows that the packet should be relayed to a subsequent SS, then it is queued for retransmission. Otherwise, it is sent to the application level.

C. MODELED PHYSICAL LAYER

To simulate the 802.16 physical layer, a modified 802.16d physical layer model was utilized. Ideally, an 802.16e physical layer model would have been used; however, a suitable 802.16e model could not be located. Instead, a model supplied in the 2010a release of MATLAB's Simulink simulation software was modified for the purposes of this thesis as shown in Figure 18. While the two physical layers are very similar, their differences have been discussed in Chapter IV. Additionally, there were several measures taken to ensure that the differences in performance were minimized and that the simulation would be a conservative estimate of actual performance.

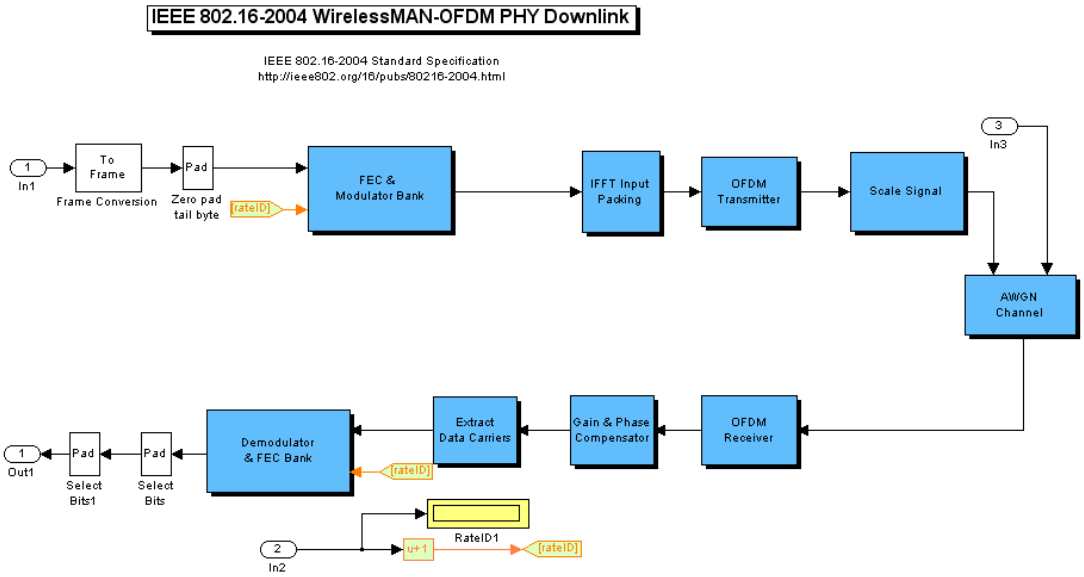

Figure 18. Modified Simulink 802.16d physical layer model used in simulation (After [43]).

1. Model Implementation

As discussed before, a major difference between the physical layers as it applies to this simulation is that the 802.16d is uses OFDM, while the 802.16e uses OFDMA. OFDM divides the transmissions up in time by giving only a single SS access to all of the data subcarriers in a channel. Thus, it allows one user access to the channel at a time and,

therefore, a form of TDMA. However, OFDMA divides up the transmissions in frequency as well as in time by giving multiple SS the use of different subcarriers in a channel at the same time. This process is called subchannelization. In addition to being a form of TDMA, OFDMA also becomes a form of FDMA. This limitation was overcome in the model through the use of multiple smaller and separate OFDM channels.

The 802.16d standard allows a single channel to only use 256 subcarriers; however, the 802.16e standard allows for scaling of the channel bandwidth between 128 carriers and 2048 carriers. For this model, a 512 and a 1024 subcarrier 802.16e channel were modeled to represent a 5 MHz and 10MHz channel, respectively. The 512 subcarrier channel was modeled using two simultaneously active 256 subcarrier 802.16d channel models, and the 1024 subcarrier channel was modeled using four simultaneously active 256 802.16d channel models. In doing so, the 802.16d channels begin to resemble the 802.16e channel, because the modeled 512 subcarrier channel can allow access to two separate SS simultaneously, and the modeled 1024 subcarrier channel can allow access to up to four separate SS simultaneously. This is shown in Figure 19.

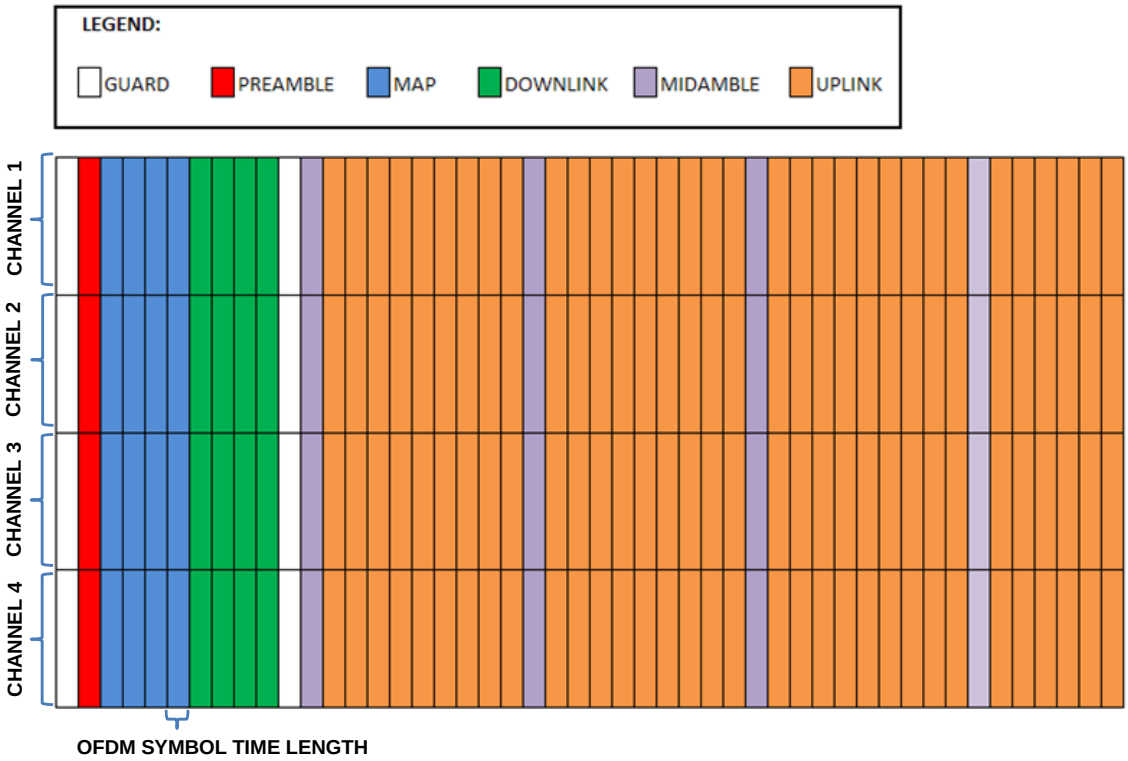

Figure 19. Modeled OFDMA frame with four individual and simultaneous channels shown in frequency and time axes.

The advantage of subchannelization is that the channel can be broken up into smaller amounts for allocations that more accurately meet the SS needs. Additionally, the allocations for one SS can be spread out in smaller increments over the entire frame transmission, so that a noise spike does not severely affect one SS transmission. Instead multiple SS transmissions may be less severely affected and the forward error correction that takes place in the 802.16 standard will be able to recover most of the data. There are multiple algorithms used by the 802.16e standard to distribute SS data throughout the entire channel; however, this data distribution feature is not instituted into this 802.16d simulation. However, data was interleaved in the model throughout the individual SS bursts to lessen the effect of noise spikes.

In both the 802.16d and the 802.16e standard, when the channel is shared between different SS, each transmission uses a coding rate appropriate to that link's signal-to-noise-plus-interference ratio (SNIR). Nevertheless, this could not be fully simulated in

Simulink because it requires a greater number of channel simulation blocks than Simulink can support. As a result, each channel was limited to a single coding rate per UL or DL frame transmission, and MAP transmissions were kept constant at BPSK. The influence of this limitation was mitigated by grouping the transmission to SS with the same or similar coding rate onto the same channel. The minimum coding rate allowable was used for the UL or DL frame transmission. Even though the coding rates were the same for all the SS on the same channel, the amount of noise added to that channel was unique to that SS's SINR.

Traditionally, the 802.16e standard is utilized for mobile SS, and the 802.16d standard is utilized for stationary SS. This is due to the improved channel estimation and synchronization qualities that were incorporated into the 802.16e standard. These qualities are partially provided for by the nondata carrying frame portions of the preamble, midamble, and guard sections as listed in Table 11. Before the MAP is transmitted by the BS, the BS transmits a preset preamble message in place of all the data subcarriers for the time period of one symbol. This signals the beginning of the DL section and is used for time and frequency estimation in addition to channel estimation [10]. However, SS mobility requires more frequent estimation updates, so a midamble is simulated to be inserted into the transmission every ten symbols to allow mobility up to 150 kmph [10]. While the preamble is actually inserted into the Simulink simulation, the midambles are simply accounted for as periods when no data is being transmitted and are compensated for later in the channel simulation calculations. A guard section that lasts one symbol period in length follows the transmission of the DL section and the UL section to signal the end of that section of the transmission.

Table 11. Modeled 802.16 frame parameters.

Parameter	Value	Reference
Time of frame	0.005s	[10]
Time of symbol	0.0001042	[10]
Subcarrier separation	9765.625 Hz	[10]
Number of subcarriers per channel	256	[10]
Number of data subcarriers per channel	192	[10]
Number of pilot subcarriers per channel	8	[10]
Number of guard and DC subcarriers per channel	56	[10]
Total number of symbols per frame	48	[10]
Number of preamble symbols per frame	1	[10]
Number of midamble symbols per frame	4	[10]
Number of guard symbols per frame	2	[10]
Number of MAP symbols per frame	4	
Number of DL symbols per frame	4	
Number of UL symbols per frame	33	

Of the 256 subcarriers that are part of the MAP, UL, or DL sections, only 192 of them carry data. The others fall into two categories: pilot subcarriers and null subcarriers. Pilot subcarriers are eight high power signals spread throughout the channel in predetermined subcarrier spaces and are used for channel estimation and channel tracking. Null subcarriers are zero power signals. One is positioned in the middle of the channel to denote the center, and the other 55 are positioned at the edges to fit the OFDM signal inside the allocated bandwidth as so to not interfere with adjacent channels [10].

The assigned ratio of OFDM symbols allocated between the UL and the DL portions is not specified in the standard, and many implementations may favor the DL over the UL. In the case of UV Sentry though, there is substantially more data flowing over the UL than the DL, so substantially more resources were allocated to the UL. The DL was allocated four symbols lengths per frame while the UL was allocated 33 symbol lengths per frame.

In the UL frame, the 802.16e standard allocates small sections for ranging and a contention-based network resource allocation request scheme. The ranging features were not implemented in this model. The resource allocation scheme is meant to allow new SS to join the network and other SS in to request more resources than they currently have. However, in the model all data requests are made via network status messages generated by the SS, and no new SS were simulated to enter the network. As previously discussed, the modulation and coding defaulted to the lowest rate necessary to meet the needs of all the users sharing the channel. Furthermore, frame resources could not be allocated as precisely as in the actual 802.16e standard. We believe that these two simulation inefficiencies more than compensate for not allocating space in the frame that would be used for these ranging and contention functions in an actual network. With the above considerations, we believe that the physical layer simulation used is a conservative approximation to an actual 802.16d radio and an even more conservative approximation to an actual 802.16e radio.

2. Physical Layer Functions

The physical layer functions are identical whether they connect the BS transmitter to the SS receiver or when they connect the SS transmitter to the BS receiver. Once a MAC burst is delivered to the physical layer for transmission, eight "0" bits are added to the end to facilitate encoding, and the data stream is sent through a punctured Reed-Solomon encoder and a convolution encoder that provides forward error correction. Afterwards, the bits are interleaved to ensure that adjacent bits are on different subcarriers and that the adjacent bits are also mapped to different places on the modulation constellation. The second portion of the interleaving is due to the fact that not all points in a modulation constellation have the same probability of error. Next, the symbols are modulated. The modulation and coding are done in accordance with what is prescribed by the BS in the MAP. The 802.16d standard prescribes that all physical layers must support, at a minimum, the modulation and coding rates given in Table 12.

Table 12. Modeled adaptive modulation and coding rates, thresholds and associated Rate IDs (After [43]).

Modulation	**Coding**	**SNIR Threshold (dB)**	**Rate ID**
BPSK	1/2	default	0
QPSK	1/2	4	1
QPSK	3/4	10	2
16 QAM	1/2	12	3
16 QAM	3/4	19	4
64 QAM	2/3	22	5
64 QAM	3/4	28	6

The above rates were included in the model. The SNIR for all the SSs receiving transmissions are evaluated, and the minimum SINR is found for the SSs. Then the minimum SINR is compared to Table 12 to find the maximum modulation and coding rate for the SINR that exceeds the SINR threshold. An example 16 QAM constellation with noise is shown in Figure 20. The accompanying rate ID above is used as a shorthand reference in the Simulink simulation for the modulation and coding pair. After the signal is modulated and coded, the nondata subcarriers (pilot and guard subcarriers) and the preamble symbols are added to the signal.

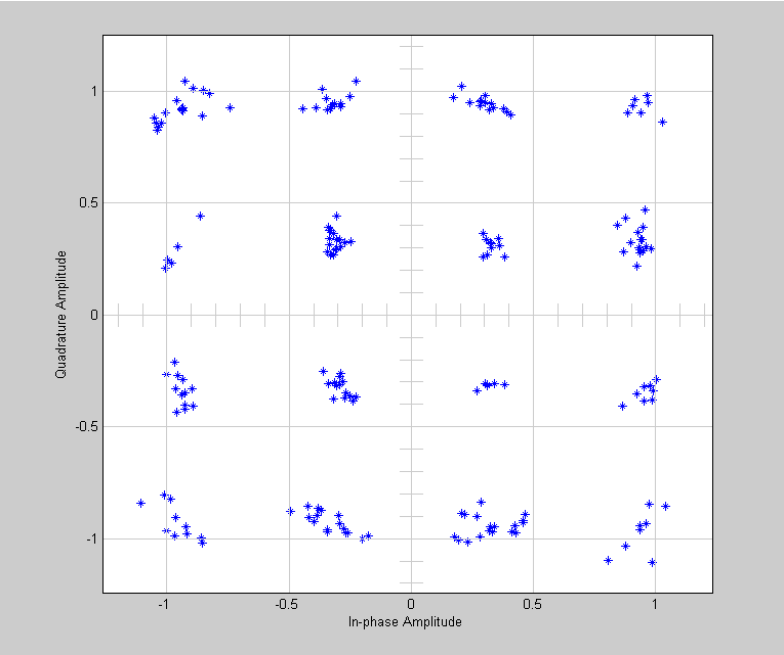

Figure 20. Example of a 16 QAM constellation with noise.

The signal is next sent through an inverse fast Fourier transform Simulink block to transform the signal from the frequency domain into the time domain before transmission. An example of a single channel's transmitted signal in the time domain is shown in Figure 21. Then, the signal transmission through a wireless medium and subsequent reception was simulated. First, fading is introduced into the signal to adjust for the multipath characteristics of the wireless medium. While an open ocean maritime environment has substantially less multipath spread than urban terrain, there is still a multipath component to the signal that is reflected from the surface of the water [52]. However, this nonline-of-sight (NLOS) signal strength was found to be a very small portion of received signal [53]. Empirical measurements in the 802.16 channel spectrum of 1.9 GHz show that this can be approximated as a Rician channel with $K = 17.6$ [54]. Accordingly, NLOS properties of the channel were modeled using a MATLAB Rician fading channel block with $K = 17.6$.

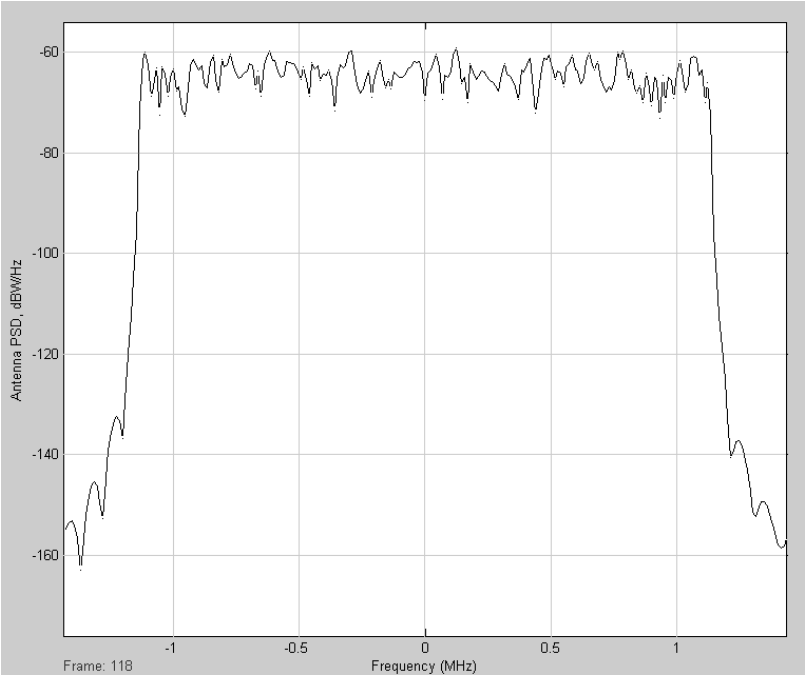

Figure 21. Example of a single channel's transmitted signal in the frequency domain.

Next, the simulated channel accounts for the assumed relative motion between the BS and the SS. Since the vehicles will most likely be transmitting FLIR video while enroute to an intercept, we assume that they will be traveling at their maximum speed of 40 kts. The relative motion between the transmitter and receiver normally introduces a substantial amount of error in the signal due to the Doppler shift of the signal. The Doppler shift causes intercarrier interference as the perceived carrier frequencies shift without the receiver being able to compensate for the shift. However, as was discussed earlier, midambles were included in the signal to allow the receiver to remain synchronized with the transmitter despite the relative motion. When midambles are introduced every 10 symbols, the receiver is able to adjust to the relative movement so that a transmitter traveling at 40 kts can be approximated by a transmitter traveling at 5.4 kts [55]. This lower perceived relative motion introduces a much more manageable amount of error. The maximum Doppler shift due to 5.4 kts was calculated to be 20.7 Hz using

$$f_{D\max} = \frac{v_{\max}}{\lambda} \quad . \tag{5}$$

This value is then used by a Simulink Rician fading channel block to model the Doppler spectrum according to the distribution introduced by [56].

The last wireless channel adjustment is the addition of noise to the signal due to thermal noise as well as interference. Because the NLOS portion of the signal is relatively small, the received power can be calculated using the free space path loss

$$\frac{P_t}{P_r} = \frac{(4\pi)^2 (d)^2}{G_r G_t \lambda^2} \tag{6}$$

as an approximation for path loss where P_t is the transmitter power, P_r is the receiver power, d is the distance between the transmitter and the receiver, G_t is the gain of the transmitter antennae, G_r is the gain of the receiver antennae, and λ is the wavelength of the carrier frequency [53]. The SINR is the ratio of the received power to the received noise and interference. The received power, noise and interference were calculated using the network parameters given in Table 13. An additive white Gaussian noise Simulink block was then used to add noise to the signal based on the calculated SINR.

Table 13. Modeled physical layer network parameters.

Parameter	**Value**	**Reference**
Carrier frequency	2.3 GHz	[10]
Complete BW	10 MHz (4 channels)	[10]
Reduced BW	5 MHz (2 channels)	[10]
BS antennae height	30 m	[10]
BS antennae radio horizon	23 km	
BS transmit power per antennae element per 10 MHz BW (3 separate sector antennas)	43 dBm	[10]
BS antennae gain	18 dBi	[10]
BS noise figure	4 dB	[10]
BS cable loss	3 dB	[10]
MS antennae height	2 m	
MS antennae radio horizon (USV)	6 km	
MS antennae height (UAV)	300 m	
MS antennae radio horizon (UAV)	unconstrained	
MS antennae gain	6 dBi	[10]
MS noise figure	4 dB	[36]
MS transmit power	40 dBm	[36]
Noise power	-174 dBm/Hz	[57]
Interference power	-174 dBm/Hz	[57]

After the signal has passed through the channel blocks, it is simulated to be received by the receiver and the preceding process is reversed. The signal is first transformed from the time domain into the frequency domain by a Simulink fast Fourier transform block, and the resulting OFDM symbols are then converted back into data. First, the nondata preamble, pilot, and guard symbols are located and removed. Second, the signal is demodulated. Third, the signal is deinterleaved. Fourth, the signal is decoded using a Viterbi decoder followed by a punctured Reed-Solomon decoder. Finally, the eight "0" tail bits at the end of the message that were initially appended for convolutional encoding at the transmitter are removed. At this point, the signal is sent by

the receiver's physical layer back up to the receiver's MAC layer and is composed of the transmitted data plus the errors introduced by the wireless channel that were not corrected by the forward error correction code.

D. NETWORK AND TRANSPORT LAYERS

For the purposes of this model, the network and transport layers were not modeled. The MAC and physical layers in the wireless network model shown in Figure 22 are governed by the 802.16 standard and modeled as explained above. The top layer is the application layer that produces and receives the transmitted traffic. However, the two layers in-between, the network and transport layers, are not necessary for the modeling of the UV Sentry System. In a larger network with many BSs, the network layer would accomplish routing functions and the transport layer would accomplish end-to-end connectivity, flow control, error control and ARQ between the numerous network nodes. However, since all messages were routed to and from the BS in this model, the network layer was unnecessary. Also, due to the simple network topology of only direct or relay links, the transport layer was also unnecessary as these functions can instead be accomplished at the MAC layer. Accordingly, the transport and network layers were ignored, and the application layer directly passed messages to and received messages from the MAC layer.

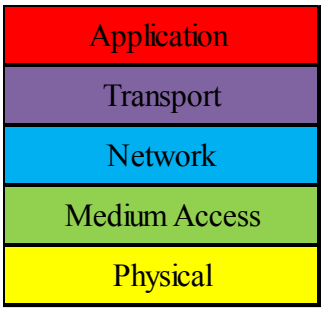

Figure 22. Wireless network model layers. The transport and network layers were not modeled for this thesis.

E. EXPERIMENT DESIGN

To evaluate the modeled network performance, a full factorial design of experiments was necessary. Five network variables were chosen to be systematically varied for the model evaluation. These variables will be referred to as factors and were chosen because we believed that they would have the greatest impact on the network performance. Each factor was assigned two levels that equate to the minimum and maximum levels most likely to be encountered in the network. While one of the factors was changed for the each experimental run, the other factors were left unchanged, and to the maximum extent possible, all outside variables were kept constant in order to best evaluate the network response to the single factor change. Each combination of factors was tested, which equates to a $2\times2\times2\times2\times2$ factorial design of experiments yielding a total of individual 32 tests. The factors chosen for selection were: 1) distance from the operational node (D), 2) presence of external interference (I), 3) number of FLIRs transmitting data to the operational node (T), 4) amount of bandwidth available to the network (S), and 5) whether the relay capability was used (R). By conducting a full factorial experiment, the impact of the above factors and the interactions between the factors is statistically analyzed in Chapter VI.

Each of the 32 tests was conducted in the same manner. Each factor was held as constant for the duration of the test. The total duration of each test in simulation time was 120 seconds; in actual time, this took approximately 8 hours to simulate in Simulink. Because we were interested in the steady state performance of the network for the experiments, the first 20 seconds of each simulation data set was not used in the subsequent analysis. This left 100 seconds of network traffic data for analysis. Since a network frame is transmitted every 0.005 seconds, this resulted in 20,000 frames per channel for analysis. Also, since each FLIR produced a data packet 15 times per second, this resulted in 1500 video packets per FLIR used. So while, the length of 100 seconds may seem short, it produced a large amount of data for analysis.

Due to the modeling limitations, the DL and the UL subframes could not be tested simultaneously. Therefore, the full factorial experiment was only applied to the UL

subframes because the UL portion of the network had the most stringent QoS demands. Based on the UL results, specific DL tests were run for further analysis, and then a final network design was recommended for the UV Sentry communications network.

1. Distance Factor

Vehicle distance was varied because an increase in distance decreases the power of a transmission according to (6). Because of this, the distance of two USVs and a relay VTUAV (if the relay capability was being used) were varied to test the network performance. For a long distance transmission, the two USVs were positioned at the outer edge of the surveillance zone (at 7 km), and for a short distance transmission, the USVs were positioned at the outer edge of the engagement zone (at 2 km). The other vehicles were kept at a constant distance of 5 km for all the experiments to simulate their conducting a search for COIs. Because a relay vehicle was used for half of the experiments, there are five resulting topologies listed in Table 14 and illustrated in Figure 23, Figure 24, and Figure 25. The relay VTUAV was positioned approximately half-way between the maritime facility and the transmitting USVs in order to best optimize the SNRs between the two USV's and the relay VTUAV on the RS-SS link and between the relay VTUAV and the maritime facility on the RS-BS link. For the short distance, this VTUAV relay position was the same if interference was present or if interference was not present. For the long distance, the VTUAV relay positioned was changed depending upon if interference was or was not present. This shifting position better optimized the SNR of the relayed USVs transmissions.

Table 14. Vehicle coordinates (in km) from operating platform for different factor settings.

Vehicle		Relay:No LongD:Yes		Relay:No LongD:Yes		Relay:Yes LongD:No		Relay:Yes LongD:Yes Interfer:No		Relay:Yes LongD:Yes Interfer:Yes	
#	Type	x	y	x	y	x	y	x	y	x	y
1	VTUAV	5.00	0.00	5.00	0.00	1.00	0.00	4.50	0.00	3.50	0.00
2	VTUAV	−5.00	0.00	−5.00	0.00	−5.00	0.00	−5.00	0.00	−5.00	0.00
3	USV	0.00	−5.00	0.00	−5.00	0.00	−5.00	0.00	−5.00	0.00	−5.00
4	USV	0.00	5.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	5.00
5	USV	1.97	0.50	6.93	1.00	1.97	0.50	6.93	1.00	6.93	1.00
6	USV	1.97	−0.50	6.93	−1.00	1.97	−0.50	6.93	−1.00	6.93	−1.00

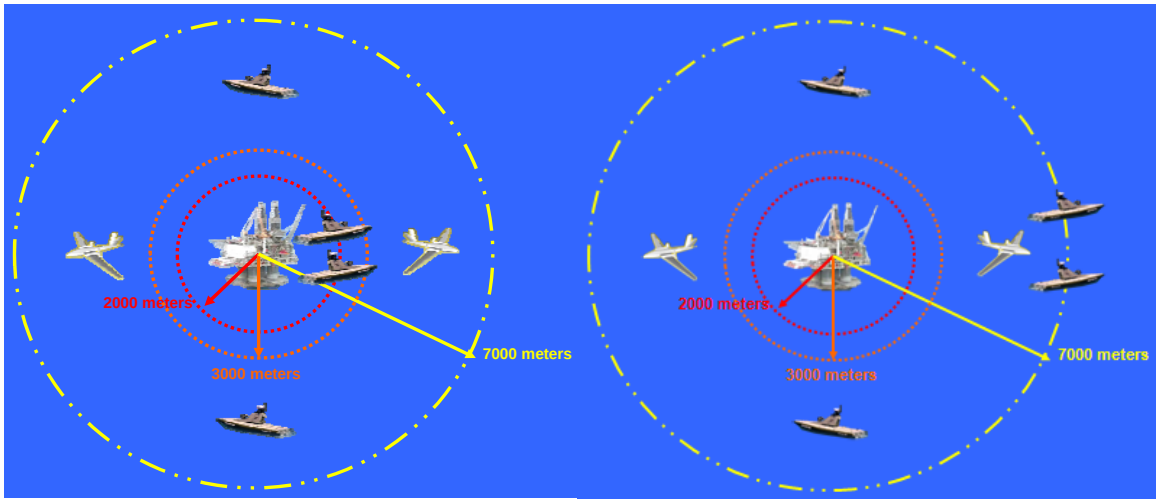

Figure 23. Short (left) and long (right) distances without relay UAV.

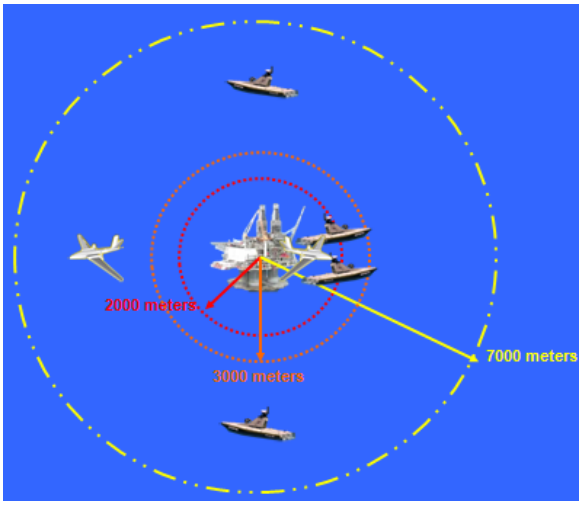

Figure 24. Short distance with relay UAV between USVs and oil platform.

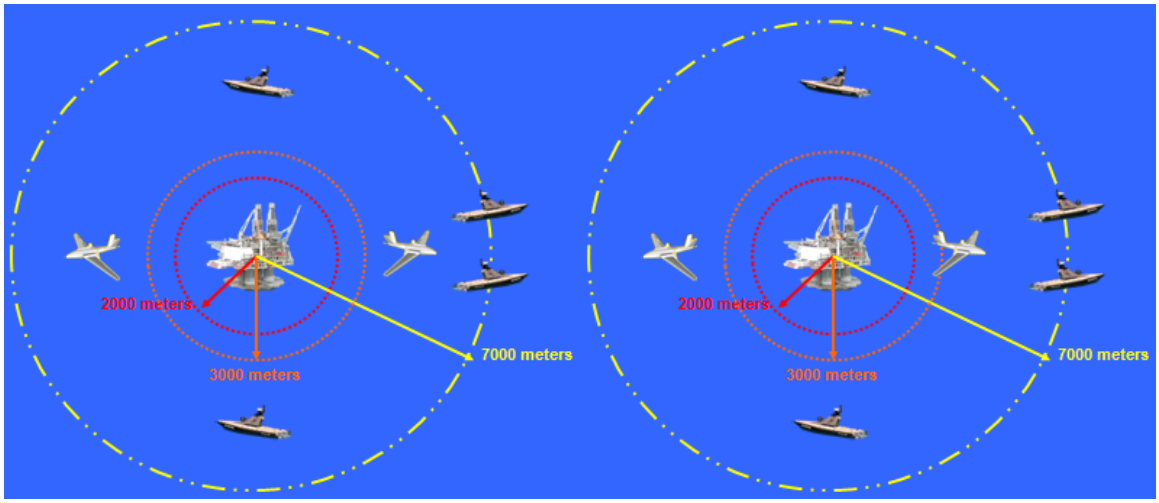

Figure 25. Long distance using a relay UAV between USVs and oil platform without interference (left) and with interference (right).

2. Interference Factor

In addition to the background noise, external interference was introduced as a factor for half of the tests. The level of interference was set to a level that was doubled the magnitude of the ambient noise. This in turn reduced the SNIR of all of the transmitted signals in the network. The interference was calculated to be the equivalent of a UV Sentry network transmitting at high power (10 Watts) on the same frequency band as the simulated network at a distance of 42 km. Thus, the interference was

introduced in order to simulate the presence of a neighboring UV Sentry network that was sharing the same frequency band but was not coordinating to share the physical medium.

3. Number of FLIR Videos Factor

In order to best test the network in a scenario where a possible threat was approaching the maritime facility, the number of uplinked FLIR NRT video streams were chosen as a factor. For half of the tests, only one FLIR video stream was transmitted, and for the other half, two FLIR video streams were transmitted. This shows how the network responds to the relatively low traffic load of one FLIR video in comparison to the higher traffic load of two FLIRs. The FLIR video streams were simulated to be transmitted from the USVs whose distances were varied, as mentioned above, to fully test the network capabilities; even though two USV's distances are varied, the one FLIR transmission case only simulates one USV as transmitting the FLIR. The network performance for the transmission of FLIR video(s) while the vehicles are close to the maritime facility and have a high SNR can be compared to the network performance for the transmissions while the vehicles are distant from the maritime facility and have a distinctly lower SNR.

All vehicles also add a baseline amount of UL traffic to the network for all the simulation runs as depicted in Table 15. All vehicles transmit network status messages to alert the operating platform (BS) of their current date requests. Also, all vehicles transmit UL control messages to coordinate with the other vehicles and also transmit that vehicle's equipment status. Furthermore, all USVs are simulated to have an operating radar that is transmitting a surface picture update. While the VTUAVs may also have sensors installed onboard, the transmission of this sensor data was not simulated. Instead, the VTUAV's primary purpose in these simulations is for relay. The VTUAVs were chosen for the relay role because their high altitude allows for a far communications horizon. If the VTUAV is in relay mode, the VTUAVs re-introduce relayed traffic to the network in addition to the network status and vehicle control traffic that originates onboard the relay.

The DL traffic simulation is simple in comparison to the UL traffic. Following the transmitted MAP, the operational platform (BS) transmits individual control messages to all the vehicles in the network. If there is a relay vehicle in use, the control message is transmitted through the relay vehicle to the intended destination.

Table 15. Network traffic transmit profiles for different UV Sentry vehicles.

Vehicle	Type	DL Control	UL Control	UL Radar	One UL FLIR Video	Two UL FLIR Video
1	VTUAV	Yes	Yes	No	No	No
2	VTUAV	Yes	Yes	No	No	No
3	USV	Yes	Yes	Yes	No	No
4	USV	Yes	Yes	Yes	No	No
5	USV	Yes	Yes	Yes	No	Yes
6	USV	Yes	Yes	Yes	Yes	Yes

4. Bandwidth Factor

The available bandwidth for the network was varied between a maximum of 10 MHz and a minimum of 5 MHz. Because the wireless network spectrum is a valuable resource and the network has the potential to be employed in a region under the control of the Federal Communications Commission or similar agency, the network should be as spectrally efficient as possible. Accordingly, if the network can meet the desired level of performance while utilizing less bandwidth, then it should. Additionally, there is the possibility that even though the network may need the full 10 MHz resources in some instances, these instances may be infrequent. The network might be able to operate most of the time using the lower bandwidth setting and, thereby, be more power efficient through the use of this "Power Saving" mode. By varying the bandwidth, we can evaluate in which instances (if any) 10 MHz is required and in which instances 5 MHz is sufficient.

5. Relay Factor

In order to evaluate the effectiveness of the relay protocol discussed earlier, it must be evaluated in comparison to the unrelayed protocol. As discussed, the advantages of the relay are expected to come at the sacrifice of metrics such as network delay and overhead. The costs in overhead and delay are due to the need of the relay to receive and decode a message before it can then recode the message for transmission on a later frame. Accordingly, the network simulations will quantify the cost of implementing a relay in this UV Sentry scenario for the different instances described above.

VI. DATA ANALYSIS

After the data was collected from the MATLAB simulations, it was used to analyze the network according to the previously discussed metrics. Statistical analysis in this chapter was completed using JMP Pro 9 software. The UL analysis was accomplished by first investigating the effect that changing the factors had on the network performance metrics. A limited DL analysis was accomplished to ensure that the DL portion of the network was capable of handling the offered traffic load under the conditions found to be the most degrading for the UL analysis. After this, we discuss the possible configurations of the network and recommended an optimal network configuration for UV Sentry. Finally, a full length simulation of the recommended network configuration was conducted in order to analyze the network performance as the vehicles move through the monitored zones in pursuit of a threatening surface vessel.

A. UL METRIC ANALYSIS

The UL network traffic consisted of four types of messages. The first was a network administrative message, the second was the vehicle control message, the third was the radar message, and the fourth was the FLIR video message. Of these four messages, the most time sensitive was the FLIR video messages. This is because a message that arrived past a certain delay threshold value was useless to the viewer of a FLIR video. Since the FLIR video messages were the most time-critical messages, it was determined that they would be the best indication of UL QoS performance. Three parameters were collected to characterize the delay of these video messages. The first was the mean delay of a FLIR packet, the second was maximum delay (for the entire simulation run) of a FLIR packet, and the third was the proportion of packets dropped (either at the video source or the destination) because of their time latency. For the above metrics, the most critical was the proportion of packets dropped because it is a clear indicator of where the network cannot meet the required QoS obligations. To analyze the network efficiency, we also analyzed the network overhead. To address reliability issues, we analyzed the SNR and related it to the measured network error rates and overall

network maximum throughput. Finally, to ensure that the control or radar packets are not too delayed, the maximum delay values for these packets are examined.

In half the simulations, only one FLIR video was simulated, and in the other half, two FLIR video simulations were simulated. In the second case, a priority scheme was set up so that the UV Sentry operator could specify a priority FLIR video stream that would have precedence over the nonpriority stream. This was accomplished so that in the event of network overload, there would be an increased chance that at least one FLIR video would be transmitted successfully. In all simulations, Vehicle 6 was modeled as always transmitting a FLIR video, and in half of the simulations, Vehicle 5 was also transmitting a FLIR video. The vehicle 6 FLIR always had priority over the Vehicle 5 FLIR. The traffic flow data for both vehicles FLIR transmission were analyzed based on data collected during the tests. While the Vehicle 6 FLIR traffic analysis could be conducted for all 32 tests based on the data listed in Table 16, the Vehicle 5 FLIR traffic analysis could only be conducted for the 16 tests listed in Table 17. This is because in the other tests, the second FLIR video was purposefully turned off. As a result, the Vehicle 5 FLIR video traffic analysis has only half the degrees of freedom to calculate the statistical significance of the factors effect.

Table 16. Priority vehicle FLIR video packet test results.

Test	Factor					Mean Delay (s)	Max. Delay (s)	Drop Rate
	L	I	T	S	R			
1	No	No	No	No	No	0.00362	0.125	0.000
2	Yes	No	No	No	No	0.00462	0.055	0.000
3	No	Yes	No	No	No	0.00519	0.050	0.000
4	Yes	Yes	No	No	No	0.00708	0.090	0.000
5	No	No	Yes	No	No	0.00369	0.135	0.000
6	Yes	No	Yes	No	No	0.00489	0.065	0.000
7	No	Yes	Yes	No	No	0.00540	0.045	0.000
8	Yes	Yes	Yes	No	No	0.00761	0.105	0.000
9	No	No	No	Yes	No	0.00518	0.125	0.000
10	Yes	No	No	Yes	No	0.00750	0.145	0.000
11	No	Yes	No	Yes	No	0.00754	0.115	0.000
12	Yes	Yes	No	Yes	No	0.01371	0.210	0.000
13	No	No	Yes	Yes	No	0.00576	0.155	0.000
14	Yes	No	Yes	Yes	No	0.00850	0.150	0.000
15	No	Yes	Yes	Yes	No	0.00900	0.135	0.000
16	Yes	Yes	Yes	Yes	No	0.01562	0.240	0.000
17	No	No	No	No	Yes	0.02170	0.145	0.000
18	Yes	No	No	No	Yes	0.02338	0.185	0.000
19	No	Yes	No	No	Yes	0.02627	0.455	0.000
20	Yes	Yes	No	No	Yes	0.03191	0.285	0.000
21	No	No	Yes	No	Yes	0.02445	0.180	0.000
22	Yes	No	Yes	No	Yes	0.02641	0.215	0.000
23	No	Yes	Yes	No	Yes	0.03038	0.555	0.000
24	Yes	Yes	Yes	No	Yes	0.03813	0.335	0.005
25	No	No	No	Yes	Yes	0.03241	0.365	0.000
26	Yes	No	No	Yes	Yes	0.03893	0.385	0.001
27	No	Yes	No	Yes	Yes	0.03550	0.515	0.004
28	Yes	Yes	No	Yes	Yes	0.06634	0.670	0.141
29	No	No	Yes	Yes	Yes	0.04070	0.400	0.009
30	Yes	No	Yes	Yes	Yes	0.05119	0.580	0.025
31	No	Yes	Yes	Yes	Yes	0.04649	0.425	0.011
32	Yes	Yes	Yes	Yes	Yes	0.08836	0.685	0.204

Table 17. Nonpriority vehicle FLIR video packet test results.

Test	Factor				Mean Delay (s)	Max. Delay (s)	Drop Rate
	L	I	S	R			
5	No	No	No	No	0.01394	0.085	0.000
6	Yes	No	No	No	0.01507	0.140	0.000
7	No	Yes	No	No	0.01093	0.105	0.000
8	Yes	Yes	No	No	0.01921	0.220	0.000
13	No	No	Yes	No	0.02161	0.215	0.000
14	Yes	No	Yes	No	0.02977	0.350	0.007
15	No	Yes	Yes	No	0.02443	0.290	0.000
16	Yes	Yes	Yes	No	0.04442	0.465	0.040
21	No	No	No	Yes	0.02594	0.260	0.000
22	Yes	No	No	Yes	0.03224	0.380	0.159
23	No	Yes	No	Yes	0.02958	0.280	0.000
24	Yes	Yes	No	Yes	0.05224	0.590	0.093
29	No	No	Yes	Yes	0.05247	0.530	0.129
30	Yes	No	Yes	Yes	0.06647	0.670	0.217
31	No	Yes	Yes	Yes	0.06303	0.680	0.159
32	Yes	Yes	Yes	Yes	0.09215	1.045	0.351

1. FLIR Mean Delay

The FLIR packet mean delay is not a complete indicator for network QoS, especially when analyzing video traffic that is highly burst-prone. As discussed before, video traffic is characterized by relatively low data-rate demands that are interspersed with spikes in traffic demand. However, in combination with other parameters, it can be useful for the purposes of comparing the network factors against each other. We first analyze the priority mean delay and then compare it to the nonpriority mean delay.

a. Priority Vehicle FLIR Mean Delay

Sorted Parameter Estimates figures similar to Figure 26 are used to depict the analysis of the network factors. The first column "Term" labels the row as either a single factor such as the "Relay" factor or the interaction between two factors such as the "Relay*SmallBW" factor interaction. The second column "Estimate" lists the factor

effect where the factor effects equate to half the values of the coefficients in a reduced regression model. For the factor effects listed in Figure 26, the reduced regression model is

$$\begin{aligned} Y = \mu + \beta_L L + \beta_I I + \beta_T T + \beta_S S + \beta_R R + \beta_{L\times I} LI + \beta_{L\times T} LT + \beta_{L\times S} LS \\ + \beta_{L\times R} LR + \beta_{I\times T} IT + \beta_{I\times S} IS + \beta_{I\times R} IR + \beta_{T\times S} TS + \beta_{T*R} TR + \beta_{S\times R} SR + e \end{aligned} \quad (7)$$

where Y is the modeled network response (in this case mean delay), μ is the modeled mean delay without any factors present, $\beta_{L,}\beta_{I,}\beta_T ... \beta_{S\times R}$ correspond to the regression coefficients (double the factor estimate values), and L, I, T, S, and R correspond to the presence of the factors where L is the long distance factor, I is the interference factor, T is the two-FLIR factor, S is the small bandwidth factor and R is the relay factor. If the factor is present, then the corresponding factor is set to 1 in (7); otherwise, it is set to 0. The parameter e represents the residual not accounted for in the model. For instance, in Figure 26, if R is present in the network, then the network mean delay is expected to increase by double the R factor effect of 0.0158633 when compared to a network where R is not present. If S is also present in the network, the mean delay is expected to increase by an amount that is double the S factor effect of 0.0065004. Also, when R and S are both present in the network, the relay times small bandwidth ($R\times S$) interaction presence will contribute an additional mean delay of double the $R\times S$ factor effect of 0.0045815.

We measured the factor effect's significance through the use of the values in the "Prob>|t|" column. These values are otherwise known as the "p-value" and indicate the probability of obtaining the calculated factor effect if the actual factor effect is zero. A p-value threshold of less than 0.05 is generally accepted as the statistically significant threshold and is used in this thesis. If a factor's p-value is less than 0.05, then the factor's effect is considered statistically significant. Otherwise, the factor effect is considered statistically insignificant and should not be distinguished from the other sources of error present in the system. The "Std Error" and "t Ratio" columns are indicators of the amount of error accounted for by the model but are not discussed further in this thesis. Next to the "Prob>|t|" column is a visual representation of the factor

effects, and the blue line indicates a value that is equal to twice the "Std Error." If the factor exceeds this blue line, then it has also exceeded the p-value of 0.05 and is considered statistically significant.

Prediction Profiler figures similar to Figure 27 are used to visually depict the absence of a single factor's effect. The dashed line at the top of the figure represents Y from (7) at its maximum value where all factors are set to 1. The effect of removing a single factor is demonstrated by comparing the "Yes" value with the "No" value for that factor. The 95% confidence interval for these predictors is shown by the blue brackets.

Interaction Profile figures similar to Figure 28 are used to visually show the interactions between the factors. To read the figure, start with the first column labeled "Relay." At the bottom of the column the axis is marked "Yes" and "No" which indicates the presence of the R factor. The blue line is the baseline case for the R factor and shows the calculated R factor effects without the presence of any other factors. This blue line is the same for all the rows in the "Relay" column. For the second row marked "SmallBW", the red line shows how the presence of S changes the effect of R. The red line and the blue line are meant to be compared to each other. If there were no interaction present, these two lines would be parallel. This is because the presence of S is seen as a constant, and R adds the same amount of mean delay to both lines as it alternates from "No" to "Yes." However, since the red and blue lines are not parallel, it indicates that the combined factors of R and S interact to increase the mean delay more than a simple additive effect.

The sorted factor estimates for the priority vehicle FLIR packet mean delay are shown in Figure 26. The Pareto chart to the right shows that R is by far the most significant factor that increases mean delay. The factor S also increases the delay, but by less than half as much as R. The presence of I and L also increase the mean delay of the packets by about a third that of R. The reason for these effects is that each factor reduces the maximum network throughput, so the likelihood of resources being allocated exactly when needed is accordingly reduced. This increases the average wait time for transmission. Lastly, while the presence of the T factor is shown to be

significant, the nonpriority FLIR transmission affects the priority FLIR transmission mean delay comparatively little when compared to the other factors. The effects of the factors on the mean delay is visually shown in Figure 26 and Figure 27, where the steeper slope indicates the stronger effect the factor's presence has on increasing the mean delay. All the network factors were shown to be statistically significant with a p-value below 0.05.

Sorted Parameter Estimates

Term	Estimate	Std Error	t Ratio	Prob>|t|
Relay[Yes]	0.0158633	0.000939	16.89	<.0001*
SmallBW[Yes]	0.0065004	0.000939	6.92	<.0001*
Relay[Yes]*SmallBW[Yes]	0.0045815	0.000939	4.88	0.0002*
Interfer[Yes]	0.004112	0.000939	4.38	0.0005*
LongDist[Yes]	0.0040912	0.000939	4.36	0.0005*
SmallBW[Yes]*LongDist[Yes]	0.0026323	0.000939	2.80	0.0128*
Relay[Yes]*LongDist[Yes]	0.0025814	0.000939	2.75	0.0143*
Relay[Yes]*Interfer[Yes]	0.0024006	0.000939	2.56	0.0212*
TwoFLIR[Yes]	0.0023662	0.000939	2.52	0.0228*
Interfer[Yes]*LongDist[Yes]	0.0023472	0.000939	2.50	0.0237*
Relay[Yes]*TwoFLIR[Yes]	0.0019885	0.000939	2.12	0.0503
SmallBW[Yes]*Interfer[Yes]	0.0016619	0.000939	1.77	0.0959
SmallBW[Yes]*TwoFLIR[Yes]	0.0012916	0.000939	1.38	0.1881
TwoFLIR[Yes]*Interfer[Yes]	0.0006004	0.000939	0.64	0.5317
TwoFLIR[Yes]*LongDist[Yes]	0.0005869	0.000939	0.62	0.5409

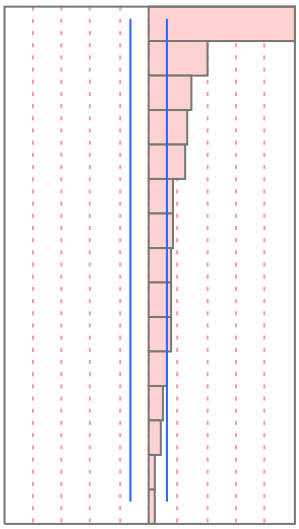

Figure 26. Sorted factor parameter estimates and p-values for priority FLIR packet mean delay (s).

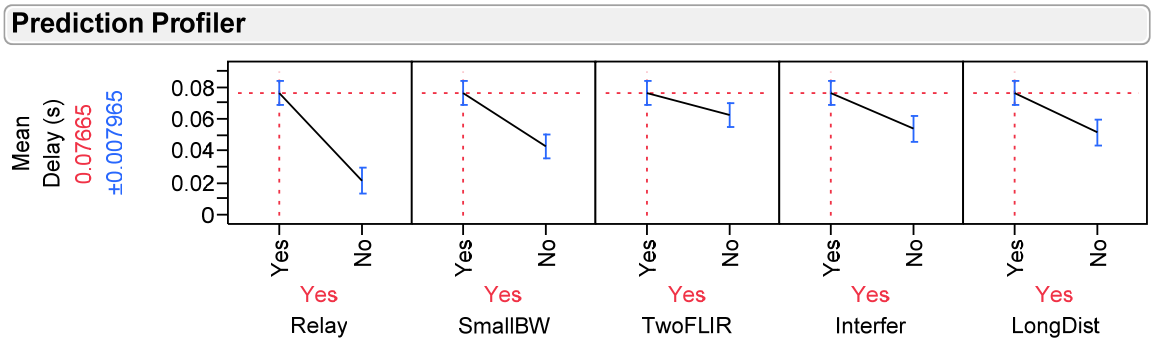

Figure 27. Response (slope endpoints) and 95% confidence interval (blue brackets) of each factor on reducing the priority FLIR packet mean delay (s) from its modeled maximum value (horizontal red dotted line).

There is significant interaction between many of the factors as shown in Figure 26. The most influential of these interactions is the $R \times S$ interaction. This interaction is greater than the individual I and L factors in order of effect on increased mean delay. This indicates that when R and S are both present in the network, the combined effect is much stronger than each factor would individually be able to contribute. Additionally, four other ($S \times L$, $R \times L$, $R \times I$, and $I \times L$) factor interactions are all shown to be statistically significant. This implies that the addition of each of the individual factors into the network has greater than simply an additive effect – the factors magnify each others effect to increase the mean delay times. This is illustrated by Figure 28 where the strongest interaction, $R \times S$, is confirmed by the large difference in slope between the two lines.

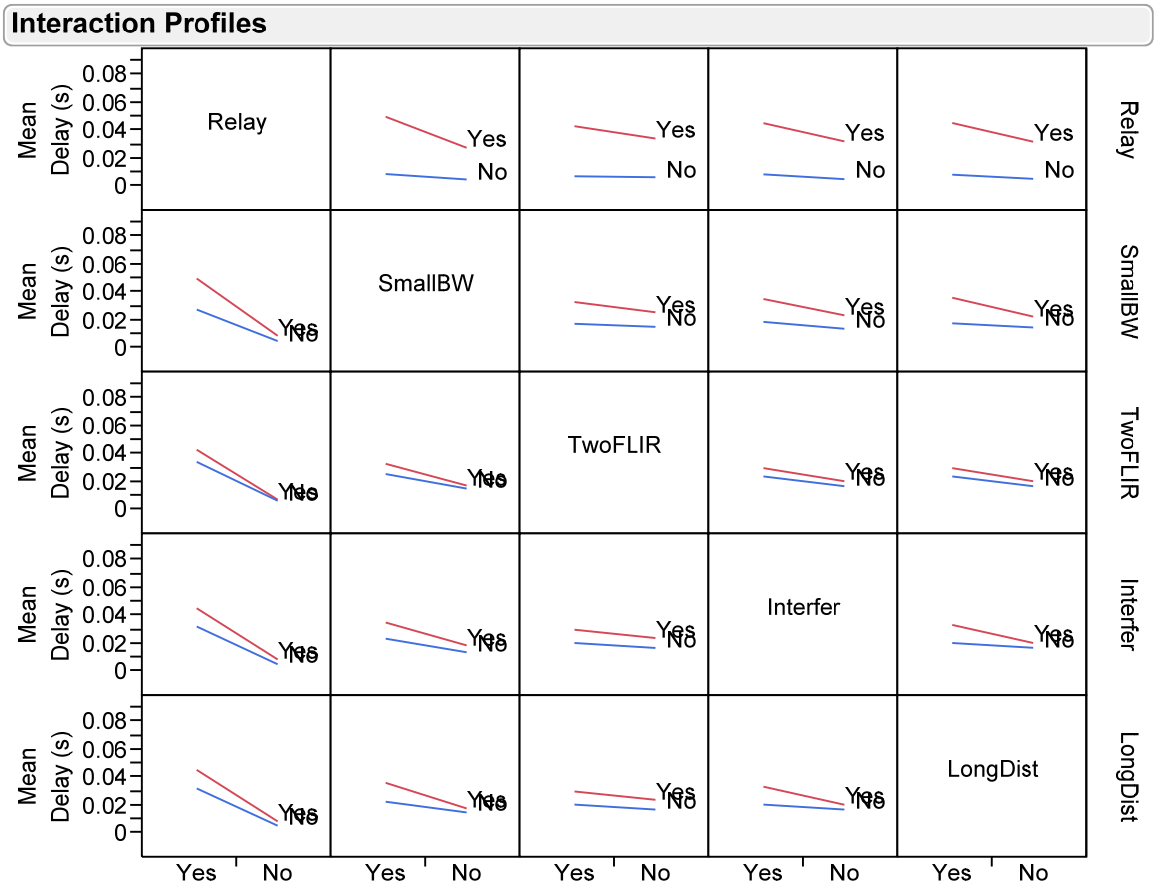

Figure 28. Factor interactions on priority FLIR packet mean delay (s).

b. Nonpriority Vehicle FLIR Mean Delay

The sorted factor estimates for the nonpriority vehicle FLIR packet mean delay are shown in Figure 29. The Pareto chart to the right shows that R is the most influential factor that increases mean delay. However, the other factors of S, L and I have a substantially greater effect on increasing mean delay for the nonpriority vehicle when compared to the priority vehicle. In fact, S increases mean relay by almost as much as R. As in the above case, the reason for these effects are that each factor reduces the likelihood of resources being allocated exactly when needed, so the average wait time for transmission increases. However, the increase in nonpriority mean delay when compared to the priority vehicle can be attributed to the priority scheme that transmits the priority FLIR traffic first. The effect of the factors on the mean delay is visually shown in Figure 30, where the steeper slope indicates the stronger effect that the factor's presence has on increasing the mean delay. The parameter T is not shown below because it is always present for the nonpriority vehicle to be transmitting FLIR video traffic. All the network factors were shown to be statistically significant.

Sorted Parameter Estimates

Term	Estimate	Std Error	t Ratio	Prob>|t|
Relay[Yes]	0.0146703	0.000431	34.03	<.0001*
SmallBW[Yes]	0.0122015	0.000431	28.30	<.0001*
LongDist[Yes]	0.0068524	0.000431	15.89	<.0001*
Interfer[Yes]	0.0049047	0.000431	11.38	<.0001*
Relay[Yes]*SmallBW[Yes]	0.0045651	0.000431	10.59	0.0001*
Interfer[Yes]*LongDist[Yes]	0.0031539	0.000431	7.32	0.0007*
Relay[Yes]*Interfer[Yes]	0.002581	0.000431	5.99	0.0019*
Relay[Yes]*LongDist[Yes]	0.0021583	0.000431	5.01	0.0041*
SmallBW[Yes]*LongDist[Yes]	0.002057	0.000431	4.77	0.0050*
SmallBW[Yes]*Interfer[Yes]	0.0018093	0.000431	4.20	0.0085*

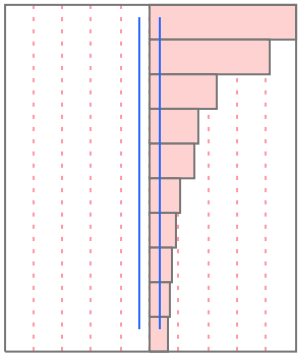

Figure 29. Sorted factor parameter estimates and p-values for nonpriority FLIR packet mean delay (s).

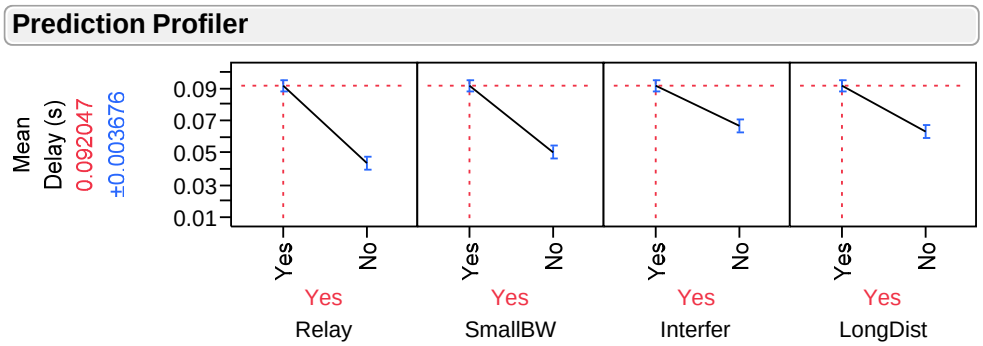

Figure 30. Response (slope endpoints) and 95% confidence interval (blue brackets) of each factor on reducing the nonpriority FLIR packet mean delay (s) from its modeled maximum value (horizontal red dotted line).

There is statistically significant interaction between all the factors as shown in Figure 29. All the interactions for the nonpriority mean delay response have as great an effect if not a greater effect than the interactions for the priority mean delay response. As in the priority case, this suggests that the addition of each of these factors into the network has greater than simply an additive effect on the network – the factors magnify each others effect to increase the mean delay times. This is illustrated by Figure 31, where the strongest interaction, $R \times S$, is confirmed by the large difference in slope between the two lines. This strong interaction can be explained by the the relay's need for additional bandwith coupled with the loss of bandwidth available when S is present, resulting in a substantially greater average wait for network resources.

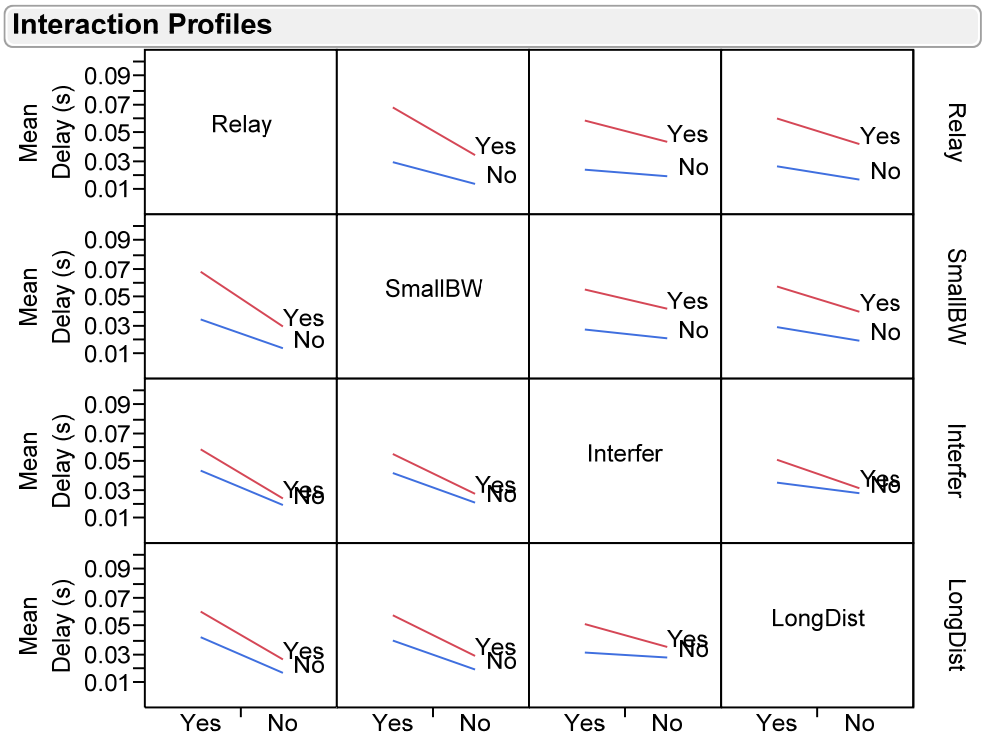

Figure 31. Factor interactions on nonpriority FLIR packet mean delay (s).

2. FLIR Maximum Delay

The maximum delay is an essential QoS metric for NRT video because the FLIR video shown at the operational platform operates at a specified delay from the video transmitter. Accordingly, any packets that arrive past the specified delay threshold are useless to the video and are discarded. In order to reduce the maximum delay value, FLIR packets that are older than 0.025 seconds are discarded at the originating vehicle instead of being transmitted. However, this scheme is not a perfect solution because an exceptionally large FLIR video packet that started to be transmitted when it met the delay threshold might not be completely transmitted in one frame, and by the time it is completely transmitted it might be too old. Also, the discard feature was not implemented in the relay vehicles, so if the network is busy, nonpriority packets might be queued longer than the priority packets, thus increasing the maximum delay of these nonpriority packets in comparison to the priority packets.

a. Priority Vehicle FLIR Maximum Delay

The sorted factor estimates for the priority vehicle FLIR packet maximum delay are shown in Figure 32. The Pareto chart to the right shows that R is followed closely by S as the most influential factors that increase maximum delay. To a lesser extent, the presence of L and I also increase the maximum delay. Lastly, while the presence of T is shown to be significant, the nonpriority FLIR transmission affects the priority FLIR transmission maximum delay comparatively less than the other factors. The reason for these effects is that each factor reduces the likelihood of resources being allocated exactly when needed, so the maximum wait time for transmission increases. The effects of the factors on the maximum delay is visually shown in Figure 33, where the steeper slope indicates the stronger effect the factor's presence has on increasing the maximum delay. All the network factors were shown to be statistically significant.

Sorted Parameter Estimates

Term	Estimate	Std Error	t Ratio	Prob>|t|
Relay[Yes]	0.1165625	0.006149	18.96	<.0001*
SmallBW[Yes]	0.093125	0.006149	15.14	<.0001*
Relay[Yes]*SmallBW[Yes]	0.0553125	0.006149	9.00	<.0001*
LongDist[Yes]	0.036875	0.006149	6.00	<.0001*
Interfer[Yes]*LongDist[Yes]	0.0275	0.006149	4.47	0.0004*
Relay[Yes]*LongDist[Yes]	0.0259375	0.006149	4.22	0.0007*
Interfer[Yes]	0.025	0.006149	4.07	0.0009*
Relay[Yes]*Interfer[Yes]	0.0228125	0.006149	3.71	0.0019*
SmallBW[Yes]*Interfer[Yes]	0.018125	0.006149	2.95	0.0095*
SmallBW[Yes]*LongDist[Yes]	0.015	0.006149	2.44	0.0267*
TwoFLIR[Yes]	0.014375	0.006149	2.34	0.0327*
TwoFLIR[Yes]*LongDist[Yes]	0.0075	0.006149	1.22	0.2403
TwoFLIR[Yes]*Interfer[Yes]	-0.0075	0.006149	-1.22	0.2403
Relay[Yes]*TwoFLIR[Yes]	0.0071875	0.006149	1.17	0.2596
SmallBW[Yes]*TwoFLIR[Yes]	0.000625	0.006149	0.10	0.9203

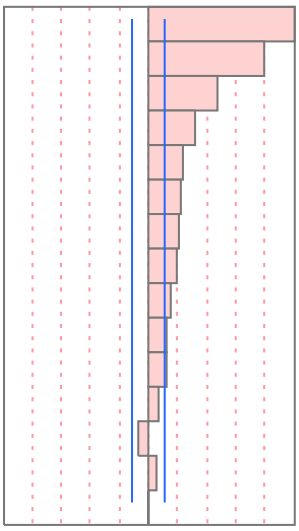

Figure 32. Sorted factor parameter estimates and p-values for priority FLIR packet maximum delay (s).

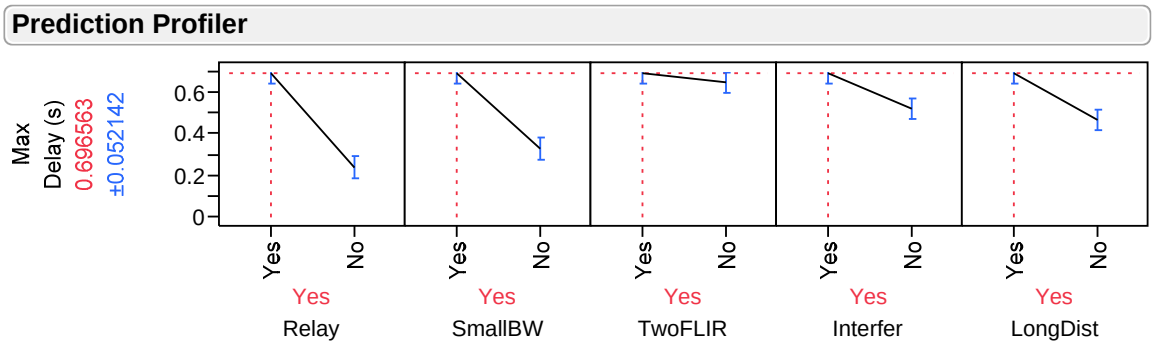

Figure 33. Response (slope endpoints) and 95% confidence interval (blue brackets) of each factor on reducing the nonpriority FLIR packet maximum delay (s) from its modeled maximum value (horizontal red dotted line).

There is significant interaction between some of the factors as shown in Figure 32. Again, the most influential of these interactions is the $R \times S$ interaction. As in the priority mean delay analysis, this interaction is greater than the I and L factors in order of impact on increased maximum delay. This indicates that the impact of this interraction is particularly strong, and when R and S are both present in the network, the combined effect is much stronger than each factor would individually be able to contribute. Additionally, five other factor interactions ($I \times L$, $R \times L$, $R \times I$, $S \times I$ and $S \times L$) are all shown to be statistically significant, which again implies that the addition of each of these factors into the network has greater than simply an additive effect on the network. This is illustrated by Figure 34 where the strongest interaction, $R \times S$, is confirmed by the large difference in slope between the two lines. Moreover, there is a statistically significant interaction between every factor except those that include the T factor.

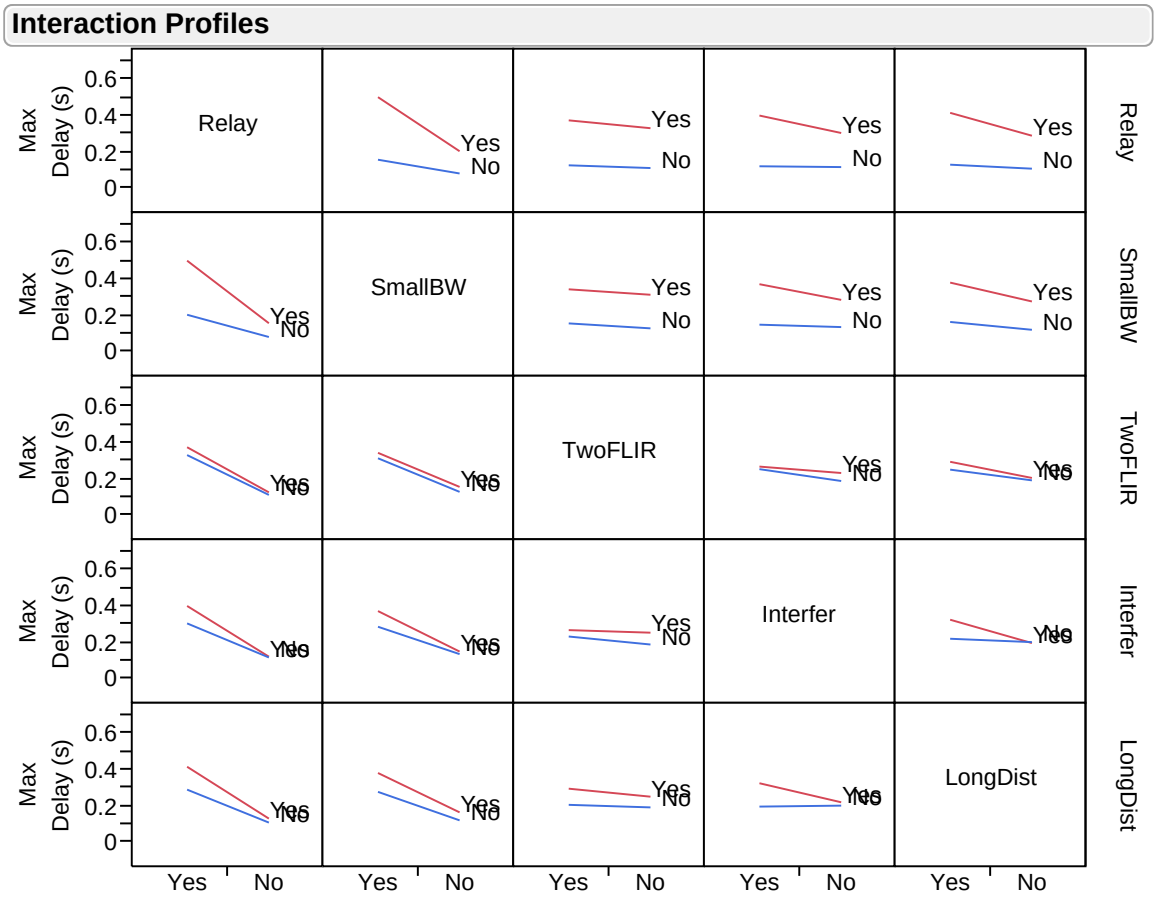

Figure 34. Factor interactions on priority FLIR packet maximum delay (s).

b. Nonpriority Vehicle FLIR Maximum Delay

The sorted factor estimates for the nonpriority vehicle FLIR packet maximum delay are shown in Figure 35. The Pareto chart to the right shows that R is the most influential factor that increases maximum delay. However, the other factors have a substantially more effect on increasing maximum delay for the nonpriority vehicle when compared to the priority vehicle. In fact, S increases maximum delay by almost as much as R. The presence of L and I also make statistically significant increases to maximum delay. This increased effect of the factors on the maximum delay when compared to the priority vehicle can be attributed to the priority scheme that transmits the priority FLIR traffic first. The factor T is not shown below because it is always present for the nonpriority vehicle to be transmitting FLIR video traffic. The effects of the

factors on the maximum delay is visually shown in Figure 36, where the steeper slope indicates the stronger effect the factor's presence has on increasing the mean delay. All the network factors were shown to be statistically significant.

Sorted Parameter Estimates

Term	Estimate	Std Error	t Ratio	Prob>|t|
Relay[Yes]	0.1603125	0.010779	14.87	<.0001*
SmallBW[Yes]	0.1365625	0.010779	12.67	<.0001*
LongDist[Yes]	0.0884375	0.010779	8.20	0.0004*
Interfer[Yes]	0.0653125	0.010779	6.06	0.0018*
Relay[Yes]*SmallBW[Yes]	0.0403125	0.010779	3.74	0.0134*
Interfer[Yes]*LongDist[Yes]	0.0321875	0.010779	2.99	0.0306*
Relay[Yes]*Interfer[Yes]	0.0290625	0.010779	2.70	0.0430*
Relay[Yes]*LongDist[Yes]	0.0284375	0.010779	2.64	0.0461*
SmallBW[Yes]*Interfer[Yes]	0.0240625	0.010779	2.23	0.0759
SmallBW[Yes]*LongDist[Yes]	0.0134375	0.010779	1.25	0.2678

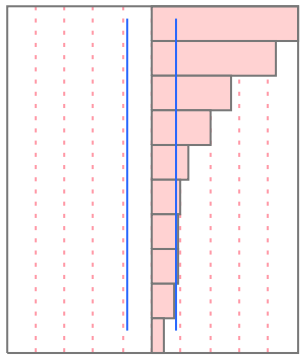

Figure 35. Sorted factor parameter estimates and p-values for nonpriority FLIR packet maximum delay (s).

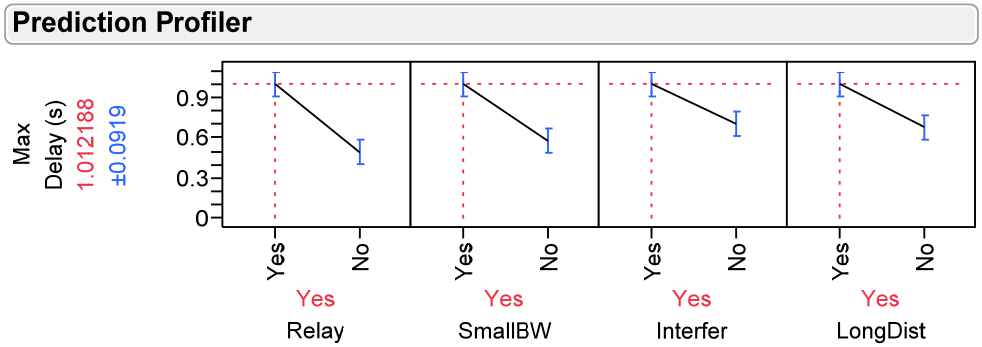

Figure 36. Response (slope endpoints) and 95% confidence interval (blue brackets) of each factor on reducing the nonpriority FLIR packet maximum delay (s) from its modeled maximum value (horizontal red dotted line).

There is statistically significant interaction between many of the factors as shown in Figure 35. Again, the interaction with the greatest effect is the $R \times S$ interaction. All the interactions for the nonpriority maximum delay response have as great an effect, if not a greatter effect, than the interactions for the priority maximum

delay response. As in the priority case, this suggests that the addition of each of these factors into the network has greater than simply an additive effect on the network; the factors magnify each others effect to increase the maximum delay times. This is illustrated by Figure 37 where again the strongest interaction, $R \times S$, is confirmed by the large difference in slope between the two lines. Again, this strong interaction can be explained by the the relay's need for additional bandwith coupled with the loss of bandwidth availabe when S is present, resulting in a substantially greater maximum wait for network resources.

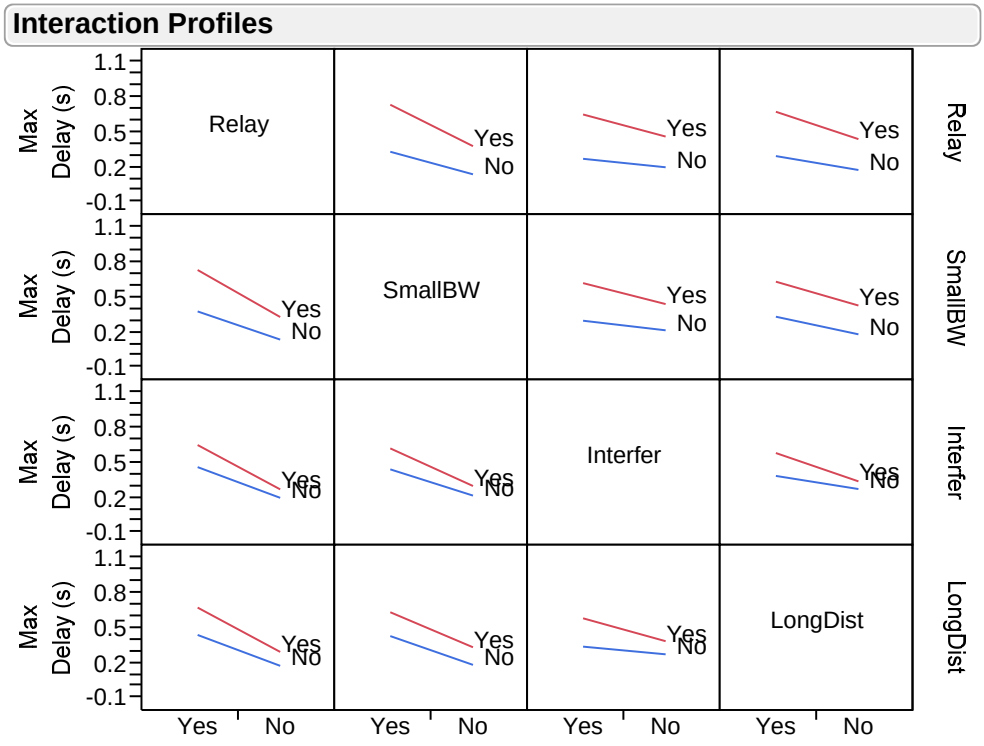

Figure 37. Factor interactions on nonpriority FLIR packet maximum delay (s).

3. FLIR Packets Dropped

As discussed above, FLIR video packets are dropped at the originating vehicle if they become older than 0.025 seconds and have not begun to be transmitted yet. However, this scheme is not foolproof because of the delays that can be introduced in

large packet transmissions and for relay delay times. The FLIR video at the operating node was set to run at a delay of 0.030 seconds. Accordingly, some packets arrived too late and are dropped at the receiver even though they were transmitted before the drop threshold of 0.025 seconds. Accordingly, the dropped packet calculations below take into account both the packets that were dropped at the transmitter because they could not be transmitted in time and those that were dropped at the receiver because they arrived too late.

a. Priority Vehicle FLIR Packets Dropped

The sorted factor estimates for the priority vehicle's proportion of FLIR packets dropped are shown in Figure 38. The Pareto chart to the right shows that R and S barely exceed the 0.05 p-value threshold and are by that definition the only significant factors that increase the number of packets dropped. The reason for these effects are that each factor reduces the likelihood of resources being allocated exactly when needed; if the priority FLIR packet queue builds up past a certain level, packets are dropped because they are no longer pertinent. While it is possible that L and I also might have some effect, they did not exceed the 0.05 threshold. Notably, the presence of T seems to have very little, if any, effect on the number of priority vehicle FLIR packets dropped. The effects of the factors on the number of packets dropped is visually shown in Figure 39, where the steeper slope indicates the stronger effect the factor's presence has on increasing the number of packets dropped. However, the effect any single factor had on increasing the number of dropped packets seems muted.

Sorted Parameter Estimates

Term	Estimate	Std Error	t Ratio	Prob>|t|
Relay[Yes]	0.0125417	0.005756	2.18	0.0446*
SmallBW[Yes]	0.0122083	0.005756	2.12	0.0499*
Relay[Yes]*SmallBW[Yes]	0.0122083	0.005756	2.12	0.0499*
LongDist[Yes]	0.0110417	0.005756	1.92	0.0731
Relay[Yes]*LongDist[Yes]	0.0110417	0.005756	1.92	0.0731
SmallBW[Yes]*LongDist[Yes]	0.0107083	0.005756	1.86	0.0813
Interfer[Yes]	0.0103333	0.005756	1.80	0.0915
Relay[Yes]*Interfer[Yes]	0.0103333	0.005756	1.80	0.0915
SmallBW[Yes]*Interfer[Yes]	0.01	0.005756	1.74	0.1015
Interfer[Yes]*LongDist[Yes]	0.0099167	0.005756	1.72	0.1042
TwoFLIR[Yes]	0.003375	0.005756	0.59	0.5658
Relay[Yes]*TwoFLIR[Yes]	0.003375	0.005756	0.59	0.5658
SmallBW[Yes]*TwoFLIR[Yes]	0.0030417	0.005756	0.53	0.6044
TwoFLIR[Yes]*LongDist[Yes]	0.002375	0.005756	0.41	0.6854
TwoFLIR[Yes]*Interfer[Yes]	0.0013333	0.005756	0.23	0.8197

Figure 38. Sorted factor parameter estimates and p-values for proportion of priority FLIR packets dropped.

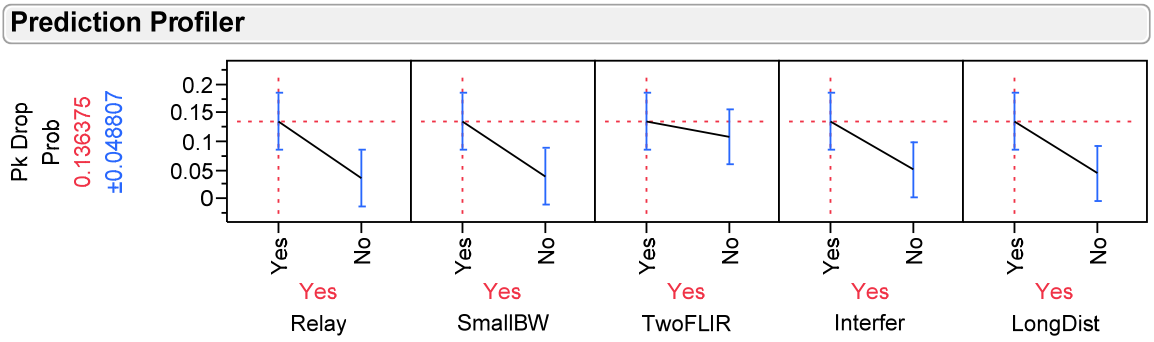

Figure 39. Response (slope endpoints) and 95% confidence interval (blue brackets) of each factor on reducing the proportion of priority FLIR packets dropped from its modeled maximum value (horizontal red dotted line).

The interactions between the factors are shown in Figure 38. The only interaction to exceed the 0.05 p-value threshold was the $R \times S$ interaction. However, several other interactions ($R \times L$, $S \times L$, $R \times I$, $S \times I$, and $I \times L$) seemed to be close to the threshold. This implies that when these factors are present in combination, their combined presence collectively contributes to the dropping of packets. This is because each factor increases the expected wait time, and if a packet is not transmitted before a

specific time threshold, it is no longer pertinent and becomes dropped. Some interaction between every factor except for those that involve T is also suggested by Figure 40. However, the minimal effect of the interactions again suggests that the priority scheme for the priority FLIR packets was mostly effective.

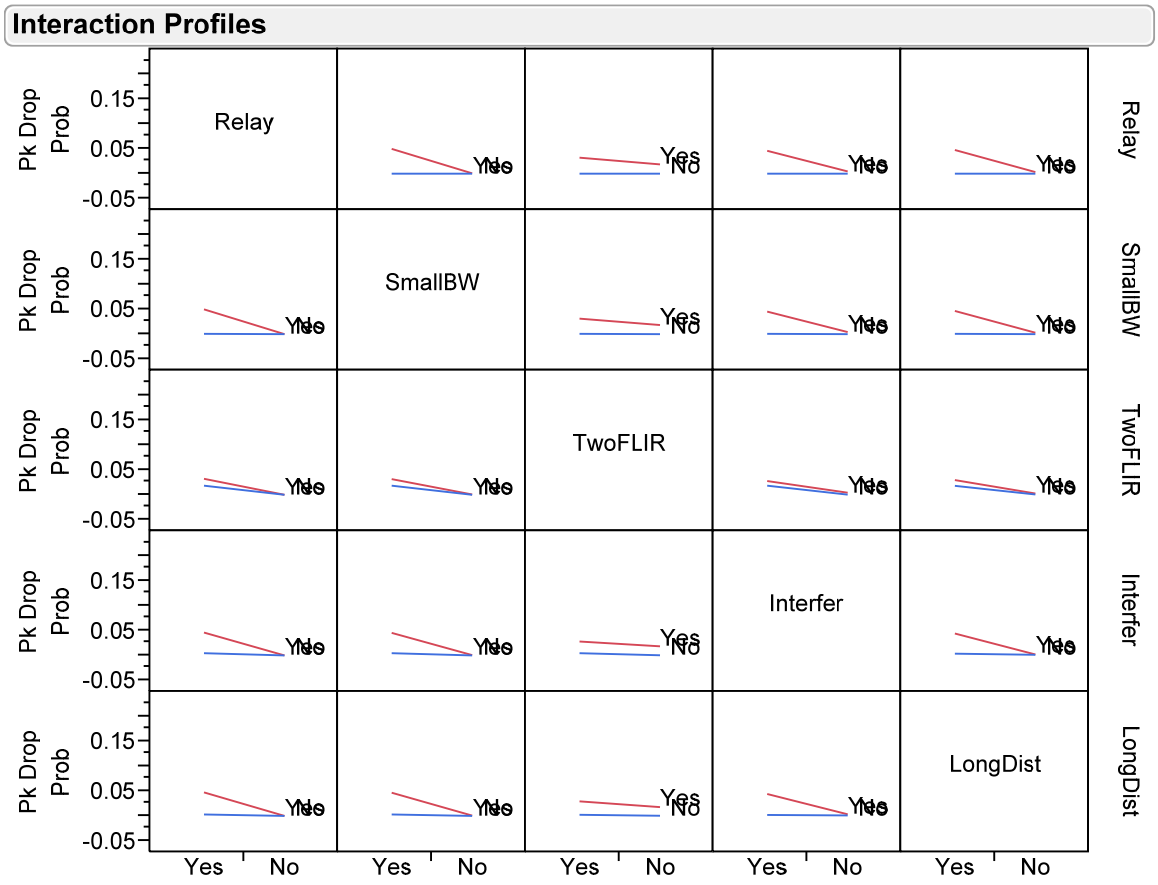

Figure 40. Factor interactions on proportion of priority FLIR packets dropped.

b. Nonpriority Vehicle FLIR Packets Dropped

The sorted factor estimates for the nonpriority vehicle's proportion of FLIR packets dropped are shown in Figure 41. Unlike in the priority vehicle case, the factors R, S, and L are all well in excess of the 0.05 threshold. As in the priority vehicle case, the reason for these effects is that each factor reduces the likelihood of

resources being allocated exactly when needed. If the nonpriority FLIR packet queue contains packets that are older than the specified threshold, the packets are dropped because they are no longer pertinent. However, the increase in number of nonpriority packets dropped when compared to the priority packets can be attributed to the priority scheme that transmits the priority FLIR traffic first. Interestingly, the factor I did not seem to noticeably affect the dropping of nonpriority FLIR packets. The factor T is not shown below because it must always be present for the nonpriority vehicle to be transmitting FLIR video traffic. The effects of the factors on the number of packets dropped is visually shown in Figure 42, where the steeper slope indicates the stronger effect the factor's presence has on increasing the number of packets dropped. All the individual factors were shown to be statistically significant.

Sorted Parameter Estimates

Term	Estimate	Std Error	t Ratio	Prob>|t|
Relay[Yes]	0.0663313	0.008847	7.50	0.0007*
SmallBW[Yes]	0.0406688	0.008847	4.60	0.0059*
LongDist[Yes]	0.0361688	0.008847	4.09	0.0095*
Relay[Yes]*SmallBW[Yes]	0.0348313	0.008847	3.94	0.0110*
Relay[Yes]*LongDist[Yes]	0.0303313	0.008847	3.43	0.0187*
SmallBW[Yes]*Interfer[Yes]	0.0165688	0.008847	1.87	0.1200
Interfer[Yes]	0.0082438	0.008847	0.93	0.3942
SmallBW[Yes]*LongDist[Yes]	0.0046688	0.008847	0.53	0.6203
Interfer[Yes]*LongDist[Yes]	0.0044188	0.008847	0.50	0.6387
Relay[Yes]*Interfer[Yes]	0.0040812	0.008847	0.46	0.6640

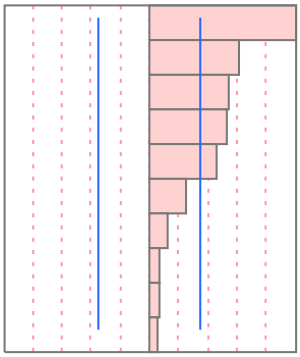

Figure 41. Sorted factor parameter estimates and p-values for proportion of nonpriority FLIR packets dropped.

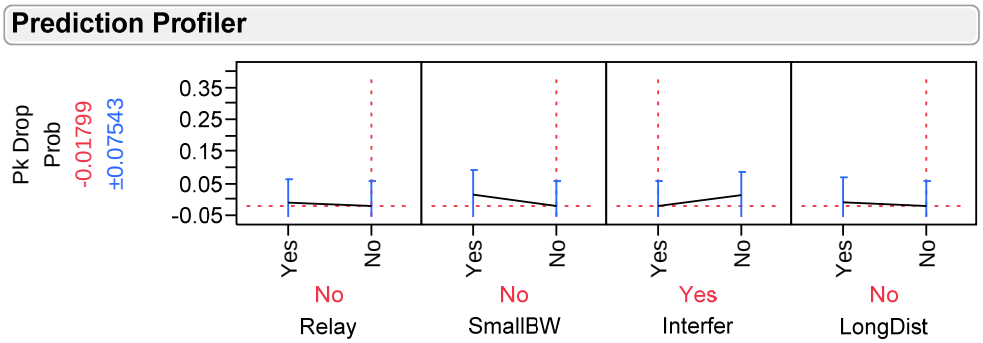

Figure 42. Response (slope endpoints) and 95% confidence interval (blue brackets) of each factor on reducing the proportion of nonpriority FLIR packets dropped from its modeled maximum value (horizontal red dotted line).

There is statistically significant interaction between the $R \times S$ and the $R \times L$ factors as shown in Figure 41. As in the priority case, the addition of each of these factors into the network has greater than simply an additive effect on the network; the factors magnify each others effect to increase the number of packets dropped. This is because each factor increases the expected wait time, and if a packet is not transmitted before a specific time threshold, it is no longer pertinent and becomes dropped. The reason the effect is more pronounced in the nonpriority case is because the priority queue has precedence, and the packets in the nonpriority queue are more likely to not be transmitted before the time threshold. This is illustrated by Figure 43 where again the strongest interactions, $R \times S$ and $R \times L$ are confirmed by the large difference in slope between the two lines.

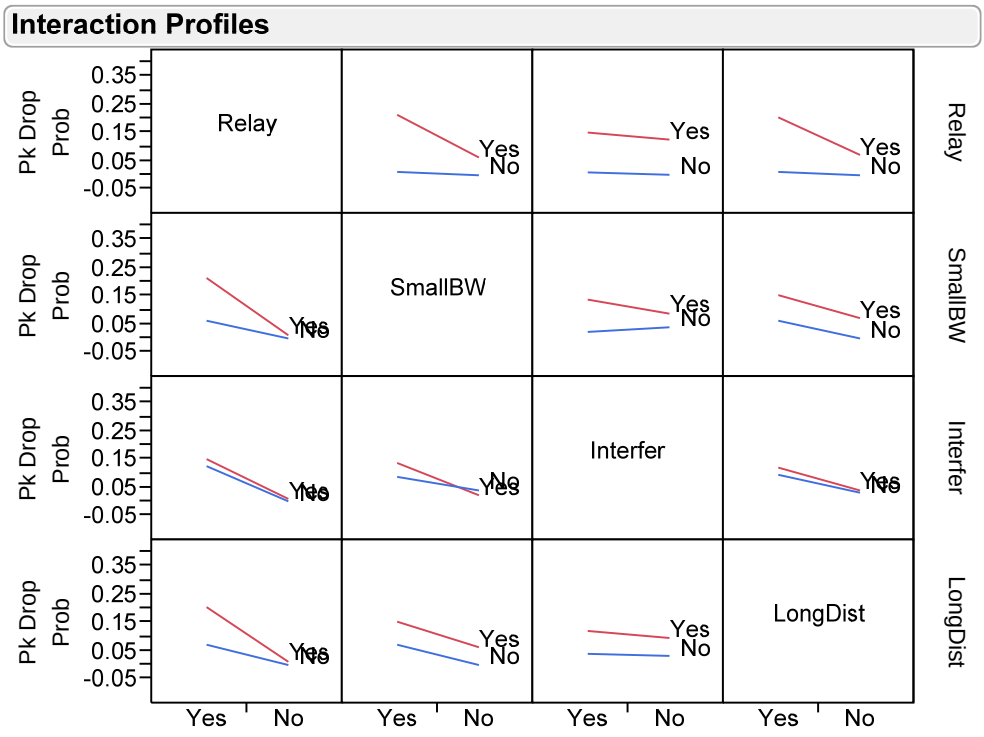

Figure 43. Factor interactions on proportion of nonpriority FLIR packets dropped.

4. Control and Radar Maximum Delay

It is also useful to examine the other packet maximum delay times to ensure that they are not taking an inordinate amount of time to be transmitted. Unlike the FLIR packets above, no control or radar packets are dropped due to delays. Accordingly, increased network utilization should correspond to an increase in transmission delay for these packet types. The maximum delay times are listed for each test in Table 18. From examining Table 18, it becomes apparent that the maximum delay for the control packets varies widely from 0.1 seconds for test 6 to 0.820 seconds for test 32. For the radar packets, the results varied from 0.074 seconds for test 5 to 0.900 seconds for test 32. While faster transmission times are preferable, given the autonomous nature of the UV Sentry control system, these maximum control delays are acceptable and should not be heavily weighted in the network configuration choice. Additionally, surface contacts do not move substantially in one second, so the slow radar contact update times are acceptable given the radar scan already takes several seconds to complete.

Table 18. Uplink control and radar packet maximum delay (s) test results.

Test	Factor					Control Max. Delay (s)	Radar Max. Delay (s)
	L	I	T	S	R		
1	No	No	No	No	No	0.175	0.200
2	Yes	No	No	No	No	0.105	0.130
3	No	Yes	No	No	No	0.145	0.170
4	Yes	Yes	No	No	No	0.190	0.215
5	No	No	Yes	No	No	0.110	0.075
6	Yes	No	Yes	No	No	0.100	0.115
7	No	Yes	Yes	No	No	0.165	0.190
8	Yes	Yes	Yes	No	No	0.225	0.250
9	No	No	No	Yes	No	0.210	0.235
10	Yes	No	No	Yes	No	0.245	0.270
11	No	Yes	No	Yes	No	0.375	0.400
12	Yes	Yes	No	Yes	No	0.450	0.475
13	No	No	Yes	Yes	No	0.240	0.265
14	Yes	No	Yes	Yes	No	0.290	0.315
15	No	Yes	Yes	Yes	No	0.415	0.440
16	Yes	Yes	Yes	Yes	No	0.540	0.565
17	No	No	No	No	Yes	0.335	0.365
18	Yes	No	No	No	Yes	0.255	0.285
19	No	Yes	No	No	Yes	0.445	0.315
20	Yes	Yes	No	No	Yes	0.350	0.375
21	No	No	Yes	No	Yes	0.335	0.365
22	Yes	No	Yes	No	Yes	0.285	0.315
23	No	Yes	Yes	No	Yes	0.585	0.335
24	Yes	Yes	Yes	No	Yes	0.480	0.410
25	No	No	No	Yes	Yes	0.435	0.465
26	Yes	No	No	Yes	Yes	0.445	0.490
27	No	Yes	No	Yes	Yes	0.675	0.705
28	Yes	Yes	No	Yes	Yes	0.750	0.810
29	No	No	Yes	Yes	Yes	0.445	0.475
30	Yes	No	Yes	Yes	Yes	0.530	0.570
31	No	Yes	Yes	Yes	Yes	0.670	0.695
32	Yes	Yes	Yes	Yes	Yes	0.820	0.900

5. Network Performance

In addition to measuring the QoS metrics for the individual service connections, it is also important to evaluate the network performance as a whole. The network throughput and goodput was measured for the network runs. These values were then compared against each other to determine the efficiency of the MAC protocols. Also, the network bit error rate and network throughput was evaluated in the context of the FLIR vehicle SNRs. The measured data is listed in Table 19.

Table 19. Overall UL network performance test results.

Test	Factor					FLIR Vehicle SNR	Bit Error Rate	Mean Throughput (bps)	Goodput to Throughput Ratio
	L	I	T	S	R				
1	No	No	No	No	No	31.18	0.00769	976500	0.888
2	Yes	No	No	No	No	20.44	0.00061	976552	0.905
3	No	Yes	No	No	No	25.08	0.00473	981632	0.900
4	Yes	Yes	No	No	No	14.34	0.00020	978465	0.904
5	No	No	Yes	No	No	31.18	0.00789	1719460	0.932
6	Yes	No	Yes	No	No	20.44	0.00059	1722207	0.942
7	No	Yes	Yes	No	No	25.08	0.00499	1722448	0.937
8	Yes	Yes	Yes	No	No	14.34	0.00019	1722027	0.941
9	No	No	No	Yes	No	31.18	0.00745	973052	0.889
10	Yes	No	No	Yes	No	20.44	0.00046	972800	0.908
11	No	Yes	No	Yes	No	25.08	0.00441	978199	0.900
12	Yes	Yes	No	Yes	No	14.34	0.00018	971835	0.908
13	No	No	Yes	Yes	No	31.18	0.00762	1714382	0.934
14	Yes	No	Yes	Yes	No	20.44	0.00056	1625811	0.959
15	No	Yes	Yes	Yes	No	25.08	0.00460	1711599	0.942
16	Yes	Yes	Yes	Yes	No	14.34	0.00018	1553456	0.958
17	No	No	No	No	Yes	27.59	0.00663	1616355	0.486
18	Yes	No	No	No	Yes	19.95	0.00479	1746589	0.488
19	No	Yes	No	No	Yes	26.16	0.00539	1702560	0.490
20	Yes	Yes	No	No	Yes	15.86	0.00012	1753877	0.491
21	No	No	Yes	No	Yes	27.59	0.00674	3082683	0.489
22	Yes	No	Yes	No	Yes	19.95	0.00454	3094064	0.487
23	No	Yes	Yes	No	Yes	26.16	0.00620	3159461	0.488
24	Yes	Yes	Yes	No	Yes	15.86	0.00012	3003018	0.495
25	No	No	No	Yes	Yes	27.59	0.00662	1596217	0.486
26	Yes	No	No	Yes	Yes	19.95	0.00464	1682854	0.488
27	No	Yes	No	Yes	Yes	26.16	0.00508	1631150	0.489
28	Yes	Yes	No	Yes	Yes	15.86	0.00009	1669051	0.493
29	No	No	Yes	Yes	Yes	27.59	0.00641	2888566	0.490
30	Yes	No	Yes	Yes	Yes	19.95	0.00449	2911700	0.490
31	No	Yes	Yes	Yes	Yes	26.16	0.00561	2946238	0.490
32	Yes	Yes	Yes	Yes	Yes	15.86	0.00010	2711489	0.497

a. Network Overhead

To evaluate the effect of factors on overhead, the ratio of goodput to overall throughput was analyzed as shown in Figure 44. Goodput is defined as the total number of bits transmitted by the application layer (such as the control, radar, or FLIR applications) that are received by the intended destination application layer. It does not include any header bits, discarded packet bits, discarded burst bits, padding bits and only counts the relayed bits once as they reach the final destination application layer. Throughput is defined as the total number of bits transmitted in the network. The use of a relay in the network dramatically increases the overhead when compared to the other factors as shown in Figure 45. As opposed to the network that did not use relays, the relayed data bits were counted as overhead. Since approximately half the bits in a relayed link are automatically counted as overhead for a two-hop relay, this high amount of overhead is unsurprising. The factor T seems to reduce overhead, which is also unsurprising since the FLIR packets tend to be large, so the ratio of header data to application data is reduced. The factors L, I, and S also seemed to reduce overhead. All of these factors place restrictions on the network, forcing it to send the same amount of data through a smaller available pipe. Accordingly, the network nodes respond by clumping the data they need to send together in large chunks instead of small ones; thus, they reduce the header ratio. Additionally, the network nodes are less likely to be allocated more network resources than they need, so they do not send as much worthless padding data. All the network factors were shown to be statistically significant with a p-value below 0.05.

Sorted Parameter Estimates

Term	Estimate	Std Error	t Ratio	Prob>|t|
Relay[Yes]	-0.216938	0.000653	-332.3	<.0001*
TwoFLIR[Yes]	0.0118917	0.000653	18.21	<.0001*
Relay[Yes]*TwoFLIR[Yes]	-0.010108	0.000653	-15.48	<.0001*
LongDist[Yes]	0.0038688	0.000653	5.93	<.0001*
Relay[Yes]*LongDist[Yes]	-0.002617	0.000653	-4.01	0.0010*
Interfer[Yes]	0.00194	0.000653	2.97	0.0090*
SmallBW[Yes]	0.001839	0.000653	2.82	0.0124*
SmallBW[Yes]*TwoFLIR[Yes]	0.0012489	0.000653	1.91	0.0738
Relay[Yes]*SmallBW[Yes]	-0.001242	0.000653	-1.90	0.0753
SmallBW[Yes]*LongDist[Yes]	0.001188	0.000653	1.82	0.0876
Interfer[Yes]*LongDist[Yes]	-0.000665	0.000653	-1.02	0.3234
TwoFLIR[Yes]*Interfer[Yes]	-0.000382	0.000653	-0.59	0.5662
TwoFLIR[Yes]*LongDist[Yes]	0.0002836	0.000653	0.43	0.6698
Relay[Yes]*Interfer[Yes]	-0.000178	0.000653	-0.27	0.7888
SmallBW[Yes]*Interfer[Yes]	0.0000959	0.000653	0.15	0.8850

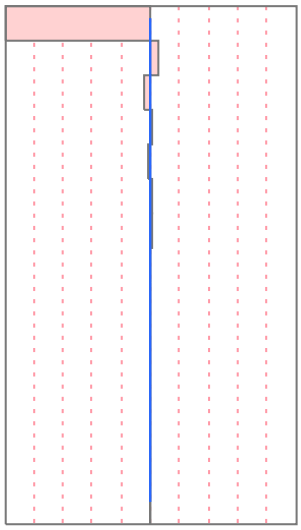

Figure 44. Sorted factor parameter estimates and p-values for proportion ratio of goodput to throughput.

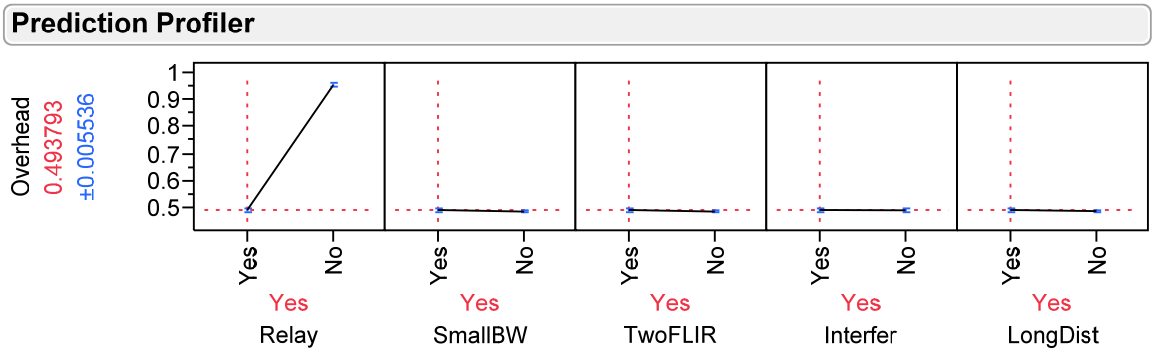

Figure 45. Response (slope endpoints) and 95% confidence interval (blue brackets) of each factor on increasing the ratio of goodput to throughput from its modeled minimum value (horizontal red dotted line).

There are two significant interactions, $R \times T$ and $R \times L$ as shown in Figure 44; however, the most influential of these interactions is the $R \times T$ interaction. The reason for this interaction is that the FLIR application is the most bandwidth intensive application modeled. When two FLIRs are being modeled to both be relayed, the proportion of relayed traffic in relation to the other traffic is increased. Since about half of this relayed traffic is counted as overhead, the overhead increases. The reason for the

$R \times L$ interaction is less clear, but at short distances the SNR for the relays are extremely good in comparison to the long distances. The increase in overhead might be due to the increase in the number of discarded packets due to corruption of the packet or burst headers. The strong $R \times T$ interaction is visible in Figure 46, but the weaker $R \times L$ interaction is difficult to see in comparison.

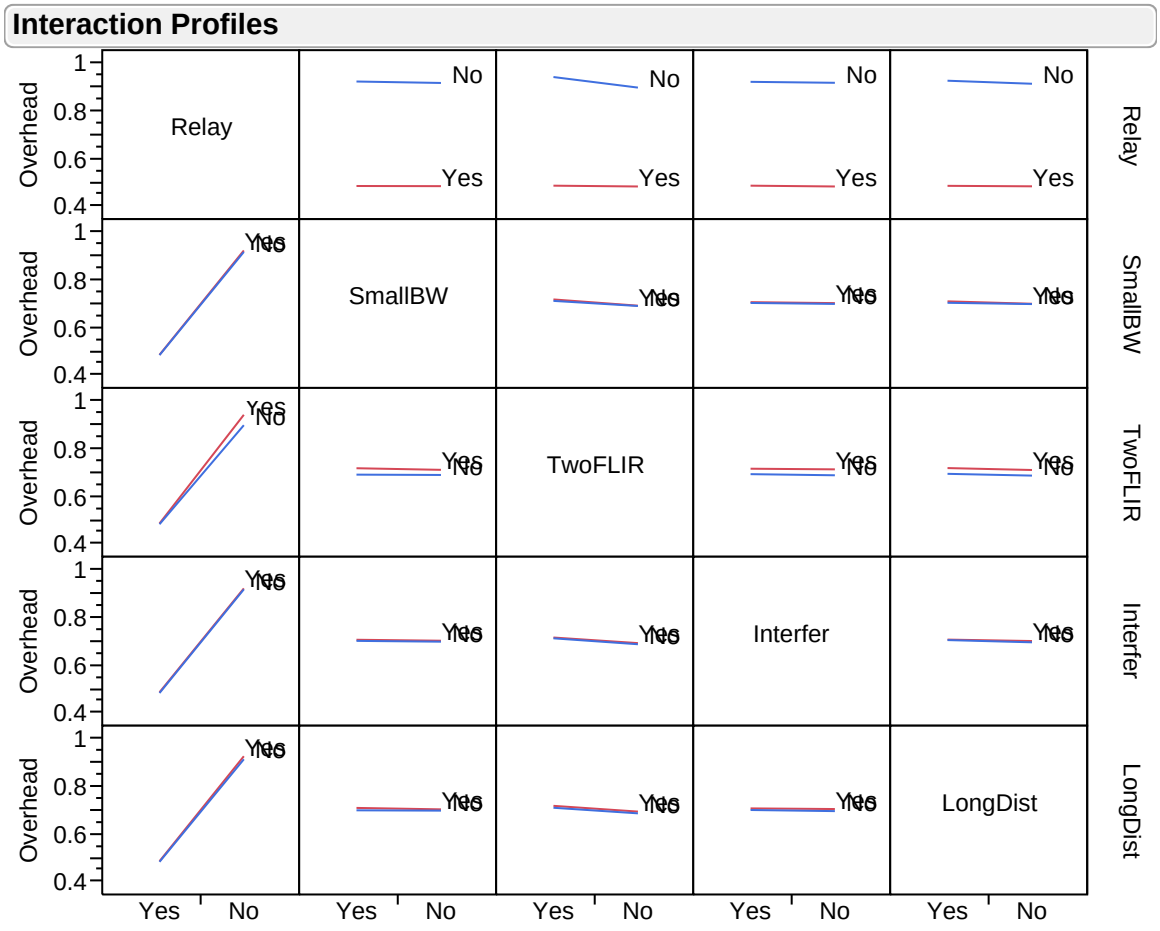

Figure 46. Factor interactions on ratio of network goodput to throughput.

b. Signal to Noise Ratio, Error Rates, and Maximum Data Rate

It is useful to analyze how the factors affect the vehicles SNR in order to understand the tradeoff between data rates and error rates. The factors with the greatest effect on SNR were L and I followed by R as a distant third. However, the reasons for

these factor effects are all related. The received power of a signal decreases with increasing distance according to equation (6), so the effect of factor L is easily explainable, and the interference is added to the ambient noise when calculating SNR, so the effect of factor I is also easily explainable. Also, one of the purposes of relays is to shorten the distance between radios and accordingly increase the SNR, so the effect of R is also explainable. The factors S and T have no impact on the SNR, so their factor values of 0 are expected. The effects of the factors on the SNR is visually shown in Figure 48, where the steeper slope indicates the stronger effect the factor's presence has on decreasing the SNR.

Sorted Parameter Estimates

Term	Estimate	Std Error	t Ratio	Prob>|t|
LongDist[Yes]	-4.9275	0.083125	-59.28	<.0001*
Interfer[Yes]	-2.215	0.083125	-26.65	<.0001*
Relay[Yes]*Interfer[Yes]	0.835	0.083125	10.05	<.0001*
Relay[Yes]*LongDist[Yes]	0.4425	0.083125	5.32	<.0001*
Interfer[Yes]*LongDist[Yes]	-0.3325	0.083125	-4.00	0.0010*
Relay[Yes]	-0.185	0.083125	-2.23	0.0408*
SmallBW[Yes]*LongDist[Yes]	2.22e-16	0.083125	0.00	1.0000
SmallBW[Yes]	0	0.083125	0.00	1.0000
TwoFLIR[Yes]	0	0.083125	0.00	1.0000
Relay[Yes]*SmallBW[Yes]	0	0.083125	0.00	1.0000
Relay[Yes]*TwoFLIR[Yes]	0	0.083125	0.00	1.0000
SmallBW[Yes]*TwoFLIR[Yes]	0	0.083125	0.00	1.0000
SmallBW[Yes]*Interfer[Yes]	0	0.083125	0.00	1.0000
TwoFLIR[Yes]*Interfer[Yes]	0	0.083125	0.00	1.0000
TwoFLIR[Yes]*LongDist[Yes]	0	0.083125	0.00	1.0000

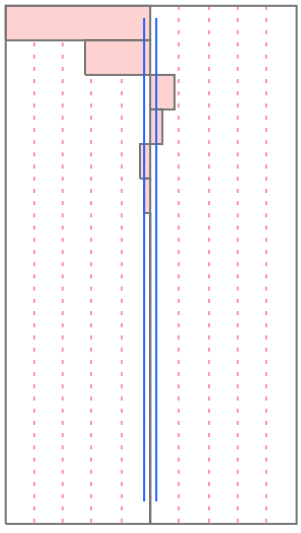

Figure 47. Sorted factor parameter estimates and p-values for FLIR vehicle radio SNR (dB).

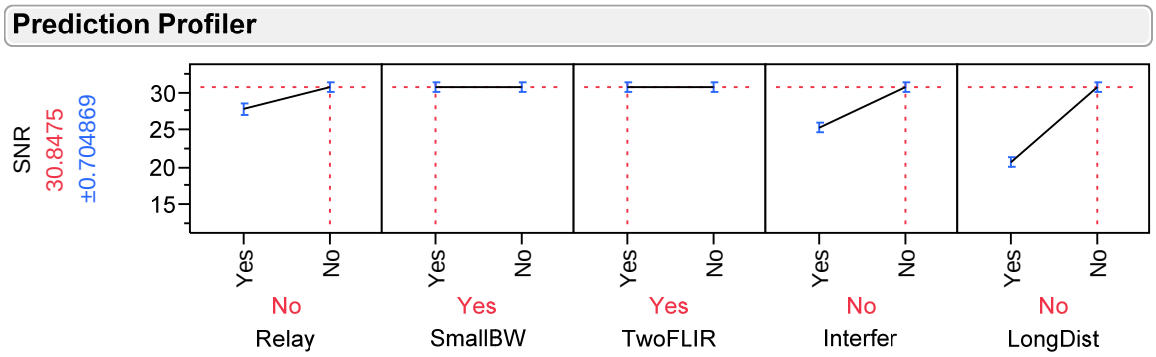

Figure 48. Response (slope endpoints) and 95% confidence interval (blue brackets) of each factor on reducing the FLIR vehicle SNR (dB) from its modeled maximum value (horizontal red dotted line).

There is significant interaction between L, I, and R as shown in Figure 47. The most influential of these interactions are the $R \times I$ and $R \times L$ interactions. These interactions are explainable due to the fact that the presence of R mitigates the SNR loss due to I and L. Additionally, the presence of I and L together magnify each other's effects to decrease SNR. Since these factor effects are fitted to a linear model and the actual transmission path losses due to distance are not linear, this interaction is expected. This is illustrated by Figure 49 where the interactions $R \times I$, $R \times L$, and $I \times L$ are confirmed by the difference in slope between the two lines. This is contrasted to the other factors' lack of interaction and parallel lines.

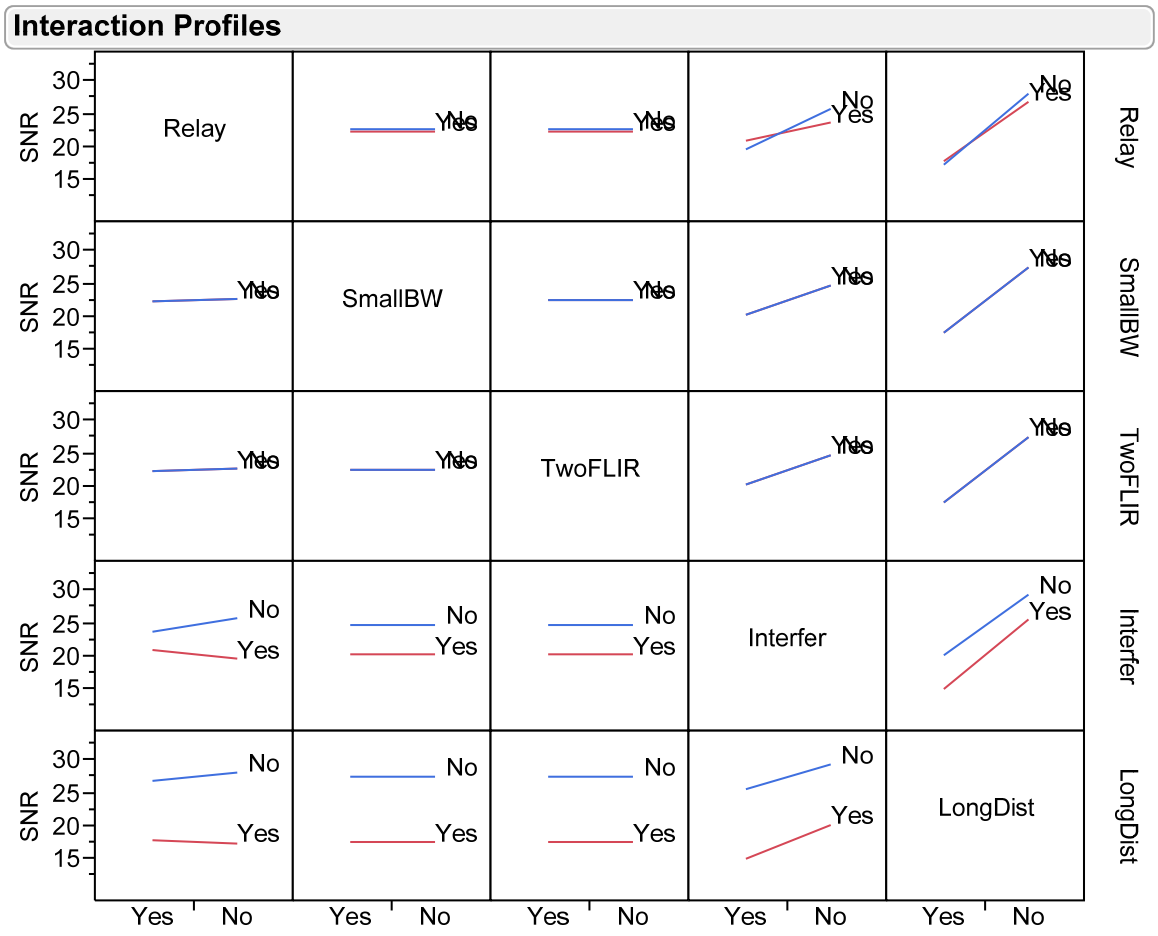

Figure 49. Factor interactions on FLIR vehicle radio SNR (dB).

The changes in SNR levels are not directly traceable to the observed error rates as seen in Table 19. This is due to the implemented AMC scheme discussed in Chapter V. If the modulation and coding rate remains at the same level, then as the SNR increases, the error rate correspondingly decreases and the reliability of the link increases. However, it is not necessary for the channel to be perfectly reliable. Instead, the modulation and coding scheme is changed when the SNR changes; this allows for an increase in link reliability to be traded for an increased data rate. The AMC mode changes were implemented so that the AMC mode increased the data throughput at certain SNR thresholds. This corresponds to the expected bit error rate jumping from a low level to a higher level once the SNR threshold is crossed. This is shown in a later network demonstration and illustrated in Figure 50. If smaller error rate jumps are

desired, more AMC modes can be implemented over the same SNR range. Additionally, if higher link reliability is desired, the SNR thresholds can be increased; however, this sacrifices the available maximum throughput data rate for increases in reliability. When the observed bit error rates for the implemented AMC SNR thresholds are examined for the vehicles transmitting FLIR video, they vary between the values of 9×10^{-5} for test 28 and 7.9×10^{-3} for test 5 as seen in Table 19. These observed bit error rates appear to be reasonable for a wireless network.

B. UL FACTOR ANALYSIS

1. Distance

Increasing the transmission distance from 2 km to 7 km was the major factor that determined the SNR for a received transmission. As the distance increased, the SNR sharply decreased and, accordingly, the maximum throughput for the network also decreased. As the maximum throughput decreased, the network had fewer resources available to allocate. As there were fewer sources available, it became more likely for the individual network nodes to request more network resources than the network could instantaneously meet. This had a moderate effect on increasing the mean delay and maximum delay as the network nodes had to wait longer to transmit their information. In both cases, increasing the distance from 2 km to 7 km increased the observed delay in the network by approximately 50%. This in turn had a relatively moderate effect on the probability of a nonpriority FLIR packet being dropped.

2. Interference

Similar to the distance factor above, the presence of an interfering transmitter had the effect of decreasing the SNR for a received transmission. The presence of interference had about half the effect of distance on decreasing SNR. Thus, it had a slightly lesser effect than distance on increasing the mean or maximum delay. Because of this lesser effect, it was not judged as a significant factor for increasing the number of

packets dropped. However, in concert with the other factors, it might have had some effect as seen when comparing tests 30 with tests 32 in Table 16 and Table 17.

3. Number of FLIR Videos

The factor of a second, nonpriority FLIR video stream barely affected the priority FLIR video stream. This factor slightly exceeded the significance threshold for mean delay and maximum delay and was, accordingly, not a significant factor for the priority packets dropped. However, in concert with the other factors, it might have had some effect as seen when comparing the proportion of packets dropped in test 28 with test 32 in Table 16.

4. Bandwidth

A decrease in bandwidth increased the mean and maximum delay for priority traffic; however, the effect on the nonpriority traffic was considerably more substantial. This is unsurprising as preference is given to the priority traffic over the nonpriority traffic. Second only to the presence of a relay, bandwidth was a very substantial factor for increasing the number of packets dropped for both priority and nonpriority traffic. Moreover, it shows a clear interaction with relay that also increases the mean delay, maximum delay, and number of packets dropped. This interaction can be explained because the use of a relay requires substantially more bandwidth because messages must be transmitted more than once.

5. Relay

The use of a relay was the overwhelming cause of delay in the network. It more than doubled the priority mean and maximum delays and almost doubled the nonpriority mean and maximum delay. Accordingly, it was shown to be the factor that had the greatest increase on both the number of priority and nonpriority packets dropped. Notably, the use of relays almost doubled the network overhead. This overhead requirement required a large increase in maximum network throughput. In exchange for this tradeoff, the presence of relays slightly increased the SNR for the vehicles

transmitting the FLIR video. Accordingly, while the relays increased the maximum throughput for this link by increasing the SNR, this increase was not enough to accommodate the large amount of excess network throughput required. This, in combination with the delay introduced in the time it took to repeat a relayed message, seemed to be the cause of the large delays and number of packet drops caused by the relays.

C. DL ANALYSIS

Unlike the UL, the UV Sentry DL that transmitted control information from the central operational platform to the UV Sentry vehicles is much simpler to analyze. There were no variable-rate radar or FLIR video transmissions transmitted on the DL. Instead there was a constant stream of control packets transmitted to all the vehicles from the operational platform. Because of this simpler network traffic profile, only a portion of the tests were necessary to analyze the DL portion. Because FLIR transmissions were not transmitted on the DL, the T factor was not pertinent to the DL and was set to a constant of "Yes." Also, the worst case scenario of long distances with interference was assumed for all tests. The factors of S and R were varied and resulted in four separate tests whose results are shown in Table 20. Under these conditions, an analysis of the DL control packet transmission delays was conducted.

Table 20. DL results for selected tests.

Test	**Factor**					**Mean Delay (s)**	**Max. Delay (s)**	**Bit Error Rate**	**Network Thrput. (bps)**	**Goodput to Thrput. ratio**
	L	I	T	S	R					
8	Yes	Yes	Yes	No	No	0.00000	0.000	0.00578	115201	0.76190
16	Yes	Yes	Yes	Yes	No	0.00000	0.000	0.00578	115201	0.76190
24	Yes	Yes	Yes	No	Yes	0.0594	0.030	0.00462	174140	0.50280
32	Yes	Yes	Yes	Yes	Yes	0.01618	0.045	0.00424	151402	0.57932

Because of the small number of tests, the tests can be compared directly to each other. The first two tests were identical because, even though the wider bandwidth was

available for use in test 16, it was not needed. These tests showed no delay before transmission because there was always enough bandwidth available to transmit this information. Tests 24 and 32 introduced a relay and, accordingly, caused a greater time delay as the packet transmissions from the operational platform required a minimum of two frames to be transmitted to the UV Sentry vehicles. This relay caused a wider deviation in transmission times because the network resources were not always immediately available as in the nonrelay case. In the presence of a relay, the control message maximum delay increased to 30 milliseconds for a 10 MHz bandwidth and, when the bandwidth was reduced to 5 MHz, the maximum delay increased to 45 milliseconds. Since the UV Sentry vehicles have a large degree of autonomy, these delays are acceptable. The bit error rates are also within an acceptable range. While the ratio of goodput to throughput may seem like it is inordinately small, it can be explained by the relatively large amounts of header in relation to the control message sizes. In summary, all configurations of the DL portion of the network appear as though they are sufficiently capable to meet the vehicle control message delay requirements of the UV Sentry vehicles.

D. NETWORK RECOMMENDATIONS

As discussed previously, the proportion of FLIR video packets dropped was the preeminent method of FLIR connection QoS characterization. Since the control packet delays, radar packet delays, and overall network performance parameters were found to be at an acceptable level for the UL, these metrics are not considered to be a limiting factor for network consideration. Because all DL configurations were shown to be sufficiently capable, the DL was also not considered to be a limiting factor for network consideration. Accordingly, the proportion of FLIR video packets dropped is used to choose the best overall network configuration.

For the factors discussed above, the distance and interference factors do not directly address the network design options, but instead are used to characterize the environment. These environmental factors drive the choice of the network design by stressing the limitations of the network. However, the factors cannot be included in the

final network configuration choice. The other factors, i.e., two FLIR videos, bandwidth size, and relay capability, directly impact the network design choice. There are a total of eight possible configurations that cover all the possible combinations of these factors. The possible configurations for UV Sentry and the assessment of the configurations are summarized in Table 21.

Table 21. Network configuration options and assessment where green denotes optimal network configuration, yellow denotes capable but sub-optimal network configuration, and red denotes overloaded network configuration.

Configuration	**Two FLIR**	**Small BW**	**Relay**	**Assessment**
1	No	No	No	Capable but not good network configuration
2	Yes	No	No	Capable but not good network configuration
3	No	Yes	No	Capable but not good network configuration
4	Yes	Yes	No	Low drop rate for nonpriority FLIR packets at long distances with and without interference
5	No	No	Yes	Capable but not good network configuration
6	Yes	No	Yes	Moderate drop rate for nonpriority FLIR packets at long distances with and without interference
7	No	Yes	Yes	High drop rate for priority FLIR packets at long distances with interference
8	Yes	Yes	Yes	High drop rate for priority FLIR packets at long distances with interference

When comparing configurations that did not use the relay capability, Configurations 1 through 3 all responded well to the above tests; however, they were not deemed as good configurations when judged against Configuration 4. Configuration 1 was not good because it had a large allocated bandwidth in relation to its small bandwidth need of transmitting one FLIR video. Configuration 2 was also deemed as not good because the network demands were still was not strained while transmitting two FLIRs. Configuration 3 was not deemed as good because, even though it only used half the

bandwidth of Configurations 1 and 2, it only transmitted one FLIR video. Configuration 4 is the best nonrelay configuration because it proved capable of transmitting two FLIR video feeds with a very low FLIR packet drop rate at long distances (with and without interference present) on a reduced bandwidth of 5 MHz.

When only comparing configurations that did use the relay capability, Configuration 6 proved to be the best configuration. Configuration 5 proved capable of transmitting a single relayed FLIR video feed in all instances; however, Configuration 6 showed that the transmission of a second FLIR video stream was possible. The one caveat to Configuration 6 was that at long distances with interference present, approximately 9% of the nonpriority FLIR vehicle packets are dropped (while the drop rate of the priority FLIR vehicle packets was only 0.5%). This drop rate is acceptable for this worst-case scenario, because as the vehicle reduces its distance from the maximum, the SNR increases and the drop rate decreases. When the bandwidth is reduced from 10 MHz to 5 MHz, the network performance decreases substantially. Configurations 7 and 8 have high drop rates on the priority vehicle FLIR at long distances when interference is present. Accordingly, they were judged as poor relay network configurations.

It is clear that for the distances examined in this thesis, a relay network is not necessary. In fact the use of a relay requires a substantially higher amount of bandwidth when compared to a nonrelay network. The nonrelay network can successfully support two FLIR videos throughout the network range on a much more power efficient 5 MHz bandwidth, but the relay network requires a 10 MHz bandwidth for a comparable performance. For this limited scenario where all the UV Sentry vehicles are within LOS range of the operational platform, the best network is Configuration 4. However, since UV Sentry is envisioned to be deployed in other scenarios that cover a wider geographical area where LOS transmissions are not guaranteed, the relay capability is a desirable mode to have available. Accordingly, a bi-modal network is suggested. For scenarios similar to the ones modeled in this thesis, a network operating in Configuration 4 is suggested; however, there are times where a vehicle might be tasked to travel outside the modeled range, possibly to investigate a suspicious vessel. In this case, if the UV

Sentry vehicles need to travel beyond the LOS transmission range of the operational platform, then the network should be able to switch modes so that it can operate in Configuration 6.

E. MODEL DEMONSTRATION

In order to demonstrate how a network operating in Configuration 6 would respond to a surface vehicle interception scenario as discussed in Chapter II, a full length simulation of a COI intercept was conducted. Two UV Sentry USVs were simulated to intercept a COI crossing into the surveillance zone at 7 km and pursue it until it enters the engagement zone at 2 km. The COI and the intercepting USVs are accordingly simulated to start at 7 km and end at 2 km. Interference is assumed to be present, both USVs are transmitting FLIR video messages to the operational facility in addition to control and radar messages, the full 10 MHz bandwidth is available, and the USVs make use of an airborne VTUAV to relay their network traffic to the operational facility. In essence, a transition between test 24 and test 23 is simulated. Additionally, two other USVs and one other VUTAV are transmitting their normal surface surveillance traffic of radar (only for USVs) and control traffic. The COI is simulated to travel at 40 kts directly towards the maritime facility, and the USVs pursue the COI also at 40 kts. The relay VTUAV maneuvers to maintain its relay position between the USVs and the operational facility. At this speed, it takes 265 seconds for the USVs to travel from a 7 km distance from the operational facility to a 2 km. The network performance for the 265 second run is shown in Figure 50 with one data point plotted for each second elapsed.

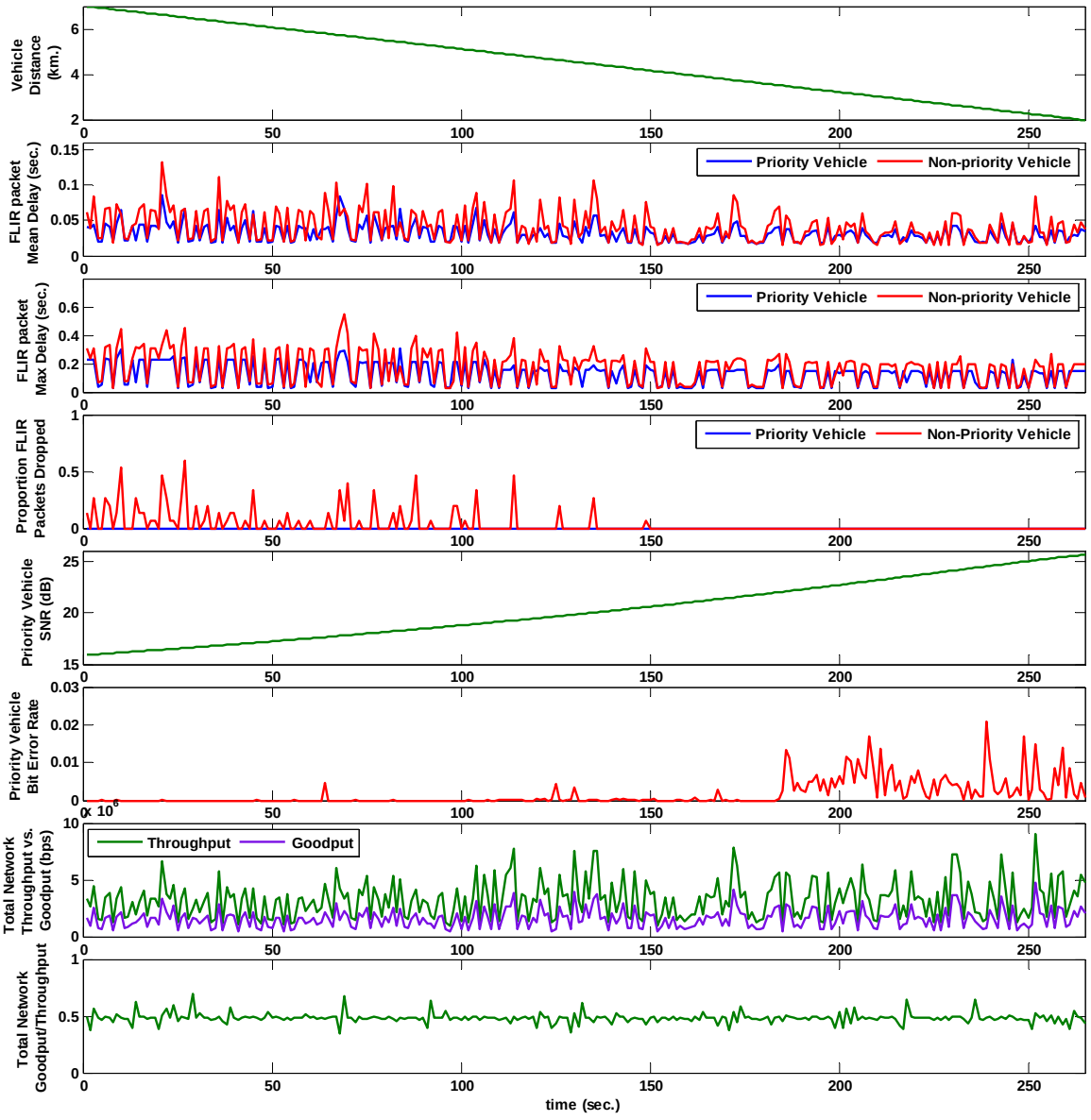

Figure 50. Recommended network configuration performance in time domain while intercepting an intruding COI.

As the USVs get closer to the operational facility, their SNR increases. As this occurs, the observed mean delay, maximum delay, and proportion of packets dropped for each time step trends lower. Nevertheless, the highly variable nature of the FLIR traffic is observable through the FLIR packet mean delay, maximum delay, and total network throughput. As the video traffic throughput spikes, there is a corresponding increase in packet mean delay, maximum delay, and drop rate. Also, a marked difference is

observable between the priority vehicle and nonpriority vehicle packet mean delay, maximum delay, and drop rate. This is due to the priority queuing given to the priority vehicle FLIR packets. Also, as the priority vehicle SNR passes the 22 dB threshold, the bit error rate is observed to increase. This is because at this point the AMC shifts the modulation/coding scheme to the next highest rate. This allows the maximum throughput to increase, but the reliability of the data decreases. The ratio of network goodput to throughput remains fairly constant, averaging at a value slightly below 0.50. In summary, the network's actual performance was similar to how it was expected to perform based on the initial analysis above.

VII. CONCLUSIONS

In this thesis we describe the design, modeling, and analysis of a communications network for UV Sentry. First, we introduced the UV Sentry system and the specific scenario that was modeled. Then by using the UV Sentry operational activities, we were able to identify the specific communications requirements for UV Sentry. Through the use of the UNTL metrics, we highlighted the metrics important to the UV Sentry operator. Then we surveyed the possible network architectures and standards in order to find the network type that best matched the UV Sentry needs; in the end we chose the 802.16 standard. Next, we examined the 802.16 standard's physical and MAC layers in detail and suggested an alternate relay implementation of the 802.16j standard. We explained the functions and parameters used in the network simulation and specified the methodology used to test the modeled network. The resulting data was first analyzed by gauging the networks response as it was measured by certain key metrics. Then we identified the most important factors for the UV Sentry network.

Finally, we recommended a network configuration that is a hybrid of configuration 4 and configuration 6 shown in Table 22. If relay capability is not required, the communications network can operate on a reduced bandwidth of 5 MHz and can still operate two separate FLIR video cameras throughout the surveillance area. However, if relay capability is desired, a 10 MHz bandwidth is required to operate two FLIR video cameras throughout the surveillance area. To more fully understand the performance of network configuration 6, a vehicle intercept scenario was conducted that used configuration 6 throughout the scenario. Two USVs intercepted a hostile threat at a distance of 7 km from the oil platform and pursued the threat until it reached a distance of 2 km from the oil platform. The resulting data were plotted, and while the network performance was acceptable at the edges of the surveillance area, the performance vastly improved as the vehicles approached the oil platform.

Table 22. Recommended UV Sentry network configurations.

Configuration	Two FLIR	Small BW	Relay	Assessment
4	Yes	Yes	No	Low drop rate for nonpriority FLIR packets at long distances with and without interference
6	Yes	No	Yes	Moderate drop rate for nonpriority FLIR packets at long distances with and without interference

A. SIGNIFICANT CONTRIBUTIONS

The significant contribution of this thesis is that a solution that meets the communications requirement for UV Sentry in the given scenario is proposed and a demonstration of an analysis method for autonomous unmanned vehicle network communications is provided. Initially the UV Sentry description and operational activities were translated into a specific scenario with defined requirements that the UV Sentry communications system must meet. The resulting network meets these requirements and also allows for future system growth. The proposed relay solution is scalable to allow for more than two hops; this allows the network coverage size to grow as the capability of UV Sentry to patrol larger geographical regions also increases. Initially, the data requirements for the UV Sentry system are envisioned to be high as the level of autonomy and trust allocated to it by the operator is small. The operator will desire to verify that the UV Sentry is making the right decisions. However, as the operator trust grows in UV Sentry and the system becomes more autonomous, the data requirements are envisioned to shrink. Without changing the communications system, the high data rate demands of UV Sentry when covering a small geographical region can later be traded for the lower data rate demand of UV Sentry that would allow it to cover a significantly larger area.

The model emulates the actual network performance at the physical and MAC layers. In many ways discussed earlier, the model built for UV Sentry in Simulink was suboptimal when compared to industry standards. Accordingly, the documented

performance of the 802.16 standard in this thesis can be assumed to be a conservative estimate of actual 802.16 performance. However, the analysis completed on the model runs allows UV Sentry system decision makers to compare the different network factors' impact on network performance and, in turn, make critical decisions about what kind of communications network UV Sentry should use.

B. RECOMMENDED FUTURE WORK

As discussed before, the modeled network is not an exact copy of an 802.16 standard. Specific abstractions and suboptimal simplifications were necessary in order to reduce the problem to a solvable level in the given timeframe. To this end, the queuing and scheduling of data transfer at the MAC layer was accomplished using simple and likely suboptimal algorithms. The only two major types of messages were modeled at the MAC layer were messages that carried application data and messages that transmitted the new network allocation requests of the SS MAC layer to BS. MAC layer functions such as ARQ and security were not modeled and might be included in future work.

Furthermore, the upper limits of Simulink's capability to compile large models were revealed throughout the simulation process, requiring a streamlining of the model, especially, at the physical layer. Instead of implementing the subchanneling as prescribed in 802.16e, multiple, concurrent and smaller channels were used. The small portions of the OFDMA frame that allow for ranging and contention between SS data allocation requests were not modeled. Further work can be spent improving on these model simplifications and adding more fidelity to the network's physical layer performance.

The use of advanced antennas onboard UV Sentry Vehicles instead of the modeled omnidirectional antennas would greatly increase network performance. A multiple-input-multiple-output system can dramatically increase data throughput and data reliability. Directional antennas as mentioned in [9] substantially increase the gain and allow for spatial reuse. While these antennas are larger in size and increase the

complexity of the radios, the implementation of these advanced antennas into UV Sentry could substantially increase the network capability and merit modeling in further studies of the UV Sentry communications network.

While the Simulink software was useful for the initial modeling of the UV Sentry system, for future work the use of an industry recognized network modeling software such as QualNet or OPNET is recommended. These software suites advertise realistic MAC and physical layer models for the IEEE 802.16 standards. Accordingly, future work might be spent using and modifying these preexisting models to ensure that the modeled network for UV Sentry closely resembles real world performance.

The modeled scenario is limited in scope, and future work can be spent working on more expanded scenarios. If a single operational platform and UV Sentry system is to cover a larger area, then the use of relays that use more than two hops can be simulated. Additionally, the handoff of vehicles between relays and the assumption or relinquishment of relay responsibility can also be modeled. However, if there is going to be more than a single UV Sentry system deployed in a region and communication between UV Sentry systems is desired, then the addition of network and transport layers to the model will be required. The network and transport layers were largely ignored because all routing was accomplished to and from the central BS. However, the addition of multiple BS in multiple UV Sentry systems may require the additional functions of discovery, routing, flow control, and end-to-end connectivity.

APPENDIX A — MODEL SOURCE CODE

Illustrated in Figure 53 and Figure 53 is a screen capture of the Simulink model used to simulate the UV Sentry communications network. The embedded MATLAB function blocks contain MATLAB source code that governs how the box translates the input into output. The source code for each embedded MATLAB function block is included below, and the complete Simulink model also available on request from the Systems Engineering department. The underlying Simulink block diagram for the channel subsystem blocks is shown in Figure 18, and the associated parameters for the Simulink blocks are contained in Chapter V. The underlying channel control subsystem is illustrated in Figure 51 and also contains an embedded MATLAB function block.

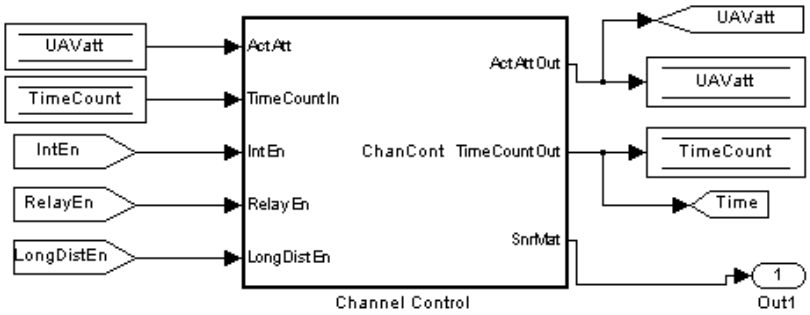

Figure 51. Channel control subsystem that calculates SNR for model.

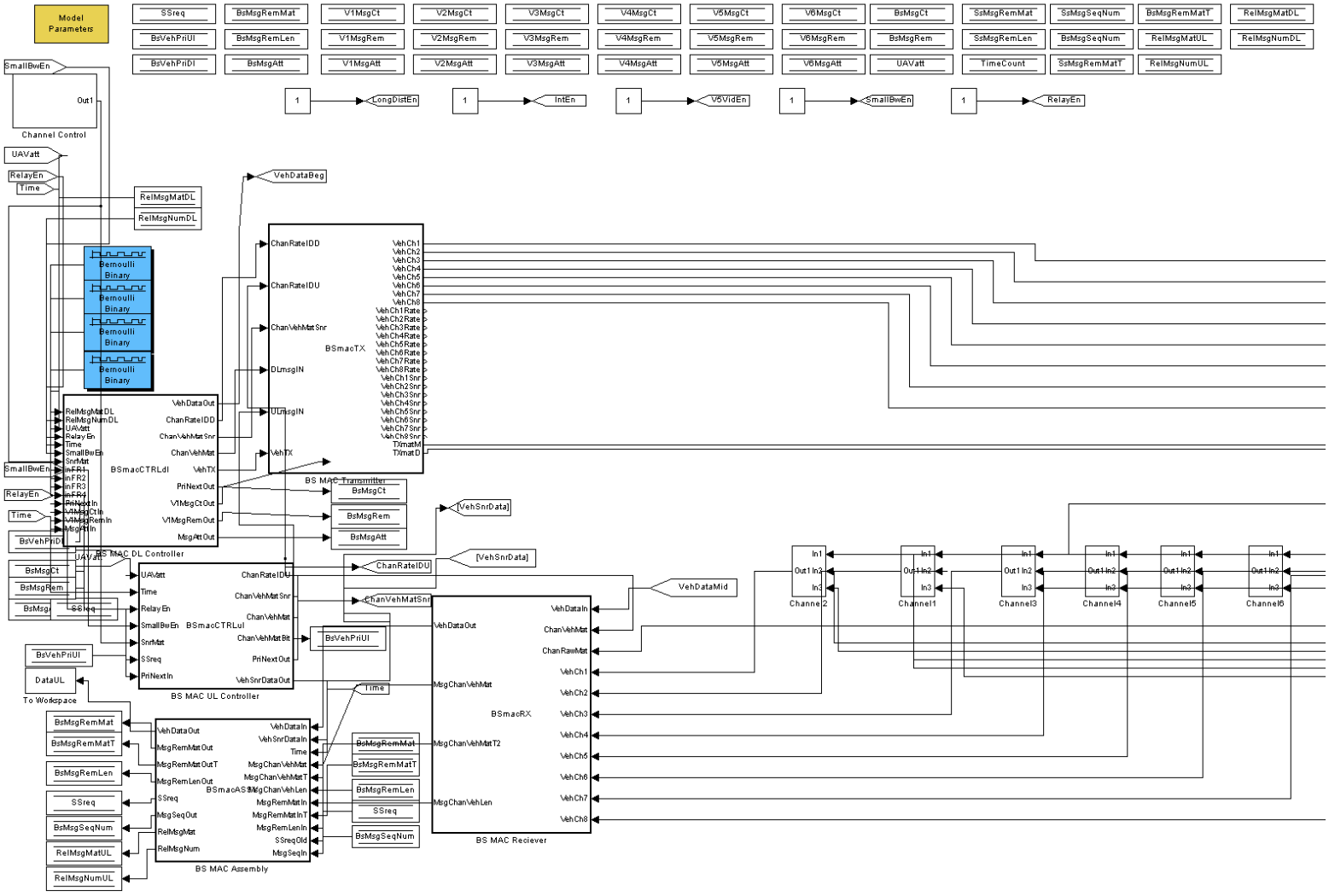

Figure 52. BS side of Simulink model simulates central operational platform.

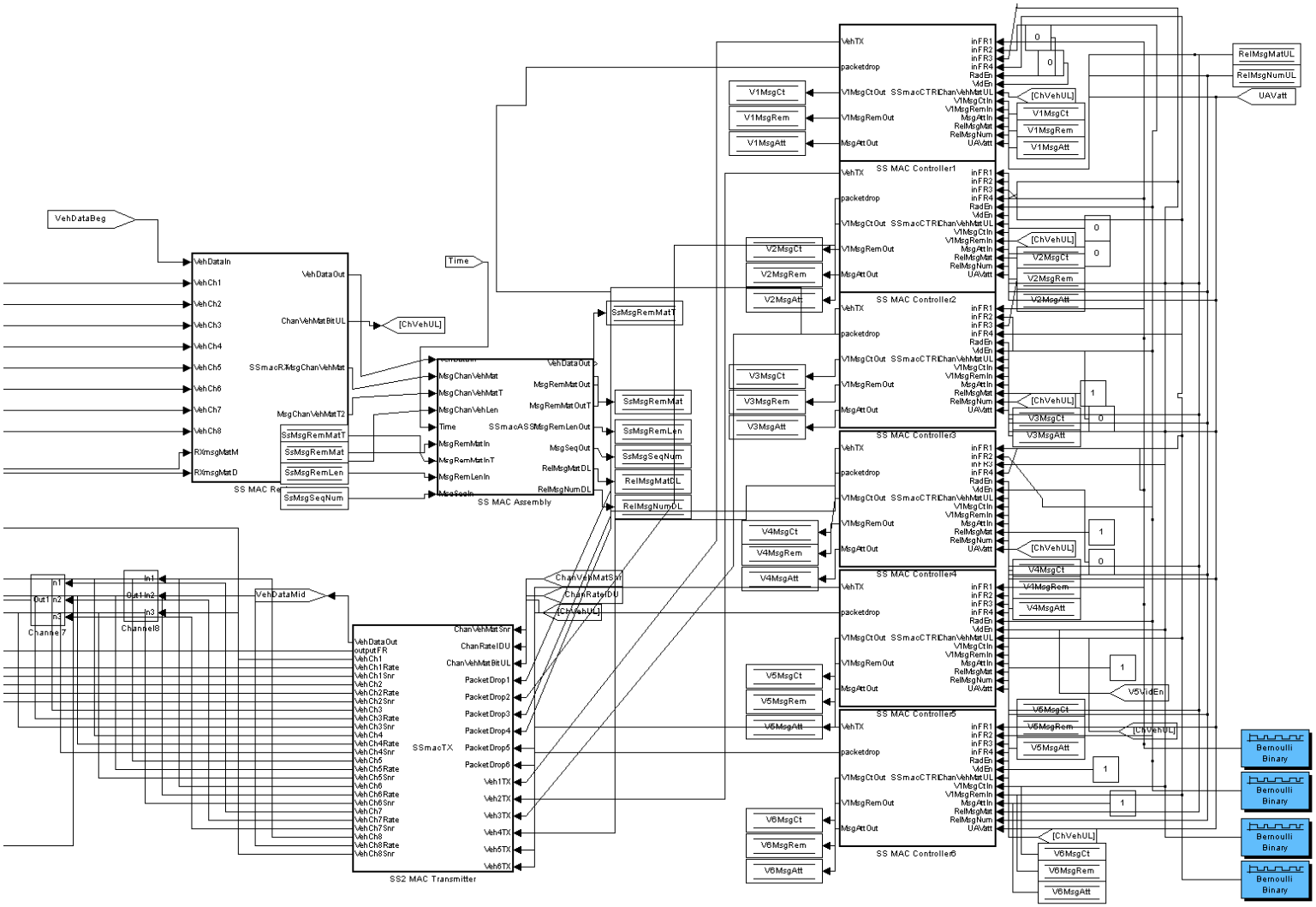

Figure 53. SS side of Simulink model with six UV Sentry vehicles modeled.

A. CHANNEL CONTROL

```
function [ActAttOut, TimeCountOut, SnrMat]  = ChanCont(ActAtt,
TimeCountIn, IntEn, RelayEn, LongDistEn)  % rate of transmission and
transmission frame size


eml.extrinsic('scatter', 'axis', 'db2pow', 'pow2db');

NumAct = 12; % number of actors
NumUAV = 10; % number of UAV's

x = 1;
y = 2;
z = 3;
hdg = 4;
spd = 5;
purs = 6;
relayedby = 7;
relaying1 = 8;
relaying2 = 9;
relaying3 = 10;

wavelength = .130435; % .130 cm = 3*10^8/(2.3*10^9); for 2.3 Ghz
BsGainDb = 18; % 18 dBi
SsGainDb = 6; % 6 dBi
NoPowDb = -174+63.9794; % -174 dBm/Hz for 10 Mhz ; 2.5 Mhz is log is
63.9794
BsTxPowDb = 43*3/4; % 3*43 dbm antennaes 20Mhz/8 chann or 10/4chan
SsTxPowDb = 40/4; % 27 dBm  40 dbm sealancet
BsNoiseFigDb = 7;% 7 db; 4 db sealancet (and book) + 3 for cable loss
SsNoiseFigDb = 4;% 8 dB; 4 db sealancet
BsNoPowDb = NoPowDb+BsNoiseFigDb;
SsNoPowDb = NoPowDb+SsNoiseFigDb;
BsGainLin = db2pow(BsGainDb);
SsGainLin = db2pow(SsGainDb);
NoPowLin = db2pow(NoPowDb);
BsTxPowLin = db2pow(BsTxPowDb);
SsTxPowLin = db2pow(SsTxPowDb);

%Interference Signal
SsIntRxPowLin = 0;
BsIntRxPowLin = 0;
SSIntRxPowDb = 0;
BSIntRxPowDb = 0;

if IntEn == 1
    SsIntRxPowLin =
(SsTxPowLin*SsGainLin*SsGainLin*wavelength^2)/(((4*pi)^2)*((42*1000)^2)
);
    SSIntRxPowDb = pow2db(SsIntRxPowLin);
    BsIntRxPowLin =
(SsTxPowLin*SsGainLin*BsGainLin*wavelength^2)/(((4*pi)^2)*((42*1000)^2)
);
```

```
    BSIntRxPowDb = pow2db(BsIntRxPowLin);
end;

BsNoiseFigLin = db2pow(BsNoiseFigDb);
SsNoiseFigLin = db2pow(SsNoiseFigDb);
BsNoPowLin = 0;
BsNoPowLin = db2pow(BsNoPowDb);
BsNoPowLin= BsNoPowLin + BsIntRxPowLin;
SsNoPowLin = 0;
SsNoPowLin = db2pow(SsNoPowDb);
SsNoPowLin = SsNoPowLin+ SsIntRxPowLin;

ActDist = zeros(NumAct,NumAct);

% all posits are relative to center in km
TimeStep = .005;  % in seconds
PolarHdg = 0;
PolarInc = 0;
CartCordHold = [0 0];

SsRxPowLin = 0;
BsRxPowLin = 0;
SsSnrLin = 0;
BsSnrLin = 0;
SsSnrDb = 0;
BsSnrDb = 0;

SnrMat = zeros(11,11);

if TimeCountIn == 0
    % place vehicles in position

    if RelayEn > 0 && LongDistEn > 0 && IntEn > 0
        ActAtt = [3.5 0 0 0 0 0 0 0 0 0; -5 0 0 0 0 0 0 0 0 0; 0 -5 0 0
0 0 0 0 0 0; 0 5 0 0 0 0 0 0 0 0; 6.93 1 0 0 0 0 0 0 0 0; 6.93 -1 0 0 0
0 0 0 0 0; 0 0 0 0 0 0 0 0 0 0; 0 0 0 0 0 0 0 0 0 0; 0 0 0 0 0 0 0 0 0
0; 0 0 0 0 0 0 0 0 0 0; 0 0 0 0 0 0 0 0 0 0; 6 8 0 0 0 0 0 0 0 0];

    elseif RelayEn > 0 && LongDistEn > 0 && IntEn == 0

        ActAtt = [4.5 0 0 0 0 0 0 0 0 0; -5 0 0 0 0 0 0 0 0 0; 0 -5 0 0
0 0 0 0 0 0; 0 5 0 0 0 0 0 0 0 0; 6.93 1 0 0 0 0 0 0 0 0; 6.93 -1 0 0 0
0 0 0 0 0; 0 0 0 0 0 0 0 0 0 0; 0 0 0 0 0 0 0 0 0 0; 0 0 0 0 0 0 0 0 0
0; 0 0 0 0 0 0 0 0 0 0; 0 0 0 0 0 0 0 0 0 0; 6 8 0 0 0 0 0 0 0 0];

    elseif RelayEn > 0 && LongDistEn == 0

        ActAtt = [1 0 0 0 0 0 0 0 0 0; -5 0 0 0 0 0 0 0 0 0; 0 -5 0 0 0
0 0 0 0 0; 0 5 0 0 0 0 0 0 0 0; 1.97 .5 0 0 0 0 0 0 0 0; 1.97 -.5 0 0 0
0 0 0 0 0; 0 0 0 0 0 0 0 0 0 0; 0 0 0 0 0 0 0 0 0 0; 0 0 0 0 0 0 0 0 0
0; 0 0 0 0 0 0 0 0 0 0; 0 0 0 0 0 0 0 0 0 0; 6 8 0 0 0 0 0 0 0 0];

    elseif RelayEn == 0 && LongDistEn > 0
        ActAtt = [5 0 0 0 0 0 0 0 0 0; -5 0 0 0 0 0 0 0 0 0; 0 -5 0 0 0
0 0 0 0 0; 0 5 0 0 0 0 0 0 0 0; 6.93 1 0 0 0 0 0 0 0 0; 6.93 -1 0 0 0 0
```

```
0 0 0 0; 0 0 0 0 0 0 0 0 0 0; 0 0 0 0 0 0 0 0 0 0; 0 0 0 0 0 0 0 0 0 0;
0 0 0 0 0 0 0 0 0 0; 0 0 0 0 0 0 0 0 0 0; 6 8 0 0 0 0 0 0 0 0];

    else

        ActAtt = [5 0 0 0 0 0 0 0 0 0; -5 0 0 0 0 0 0 0 0 0; 0 -5 0 0 0
0 0 0 0 0; 0 5 0 0 0 0 0 0 0 0; 1.97 .5 0 0 0 0 0 0 0 0; 1.97 -.5 0 0 0
0 0 0 0 0; 0 0 0 0 0 0 0 0 0 0; 0 0 0 0 0 0 0 0 0 0; 0 0 0 0 0 0 0 0 0
0; 0 0 0 0 0 0 0 0 0 0; 0 0 0 0 0 0 0 0 0 0; 6 8 0 0 0 0 0 0 0 0];

    end

    ActAtt(12,hdg) = 225-30+rand(1)*60;  % in degreees

    ActAtt(12,spd) = (10+rand(1)*40) * 0.000514444444;  %in kts
converted to kmps

    % calculates distance between vehicles
    for a1=1:NumAct
        for a2=1:NumAct

            ActDist(a1,a2) = sqrt((ActAtt(a2,1) - ActAtt(a1,1))^2 +
(ActAtt(a2,2) - ActAtt(a1,2))^2);

        end
    end

end
% allocates who relays who
if RelayEn == 1

    ActAtt(1,relaying1) = 5;
    ActAtt(1,relaying2) = 6;

    ActAtt(5,relayedby) = 1;
    ActAtt(6,relayedby) = 1;

end

TimeCountOut = TimeCountIn + 1;

% if in pursuit mode, pursues agressor
for a1 = 1:NumUAV

    ActAtt(1,purs) = 1; % one designated as pursuit

    if ActAtt(a1,purs) == 1 % if pursuit adjust position

        PolarHdg = cart2pol((ActAtt(12,x)-ActAtt(a1,x)),(ActAtt(12,y)-
ActAtt(a1,y)));  % only gives hdg

        PolarInc = ActAtt(a1,spd)*TimeStep;

        CartCordHold(1,x) = pol2cart(PolarHdg,PolarInc);
```

```
        PolarHdg = PolarHdg -pi/2;% adjust for pol2cart deffic
        if PolarHdg < 0
            PolarHdg = PolarHdg + 2*pi;
        end

        CartCordHold(1,y) = pol2cart(PolarHdg,PolarInc);

    end

end

% adjust aggressor posit
if ActAtt(12,hdg) < 90

    PolarHdg = ActAtt(12,hdg) + 270;

else
    PolarHdg = ActAtt(12,hdg) - 90;

end

PolarHdg = PolarHdg/180*pi;

PolarInc = ActAtt(12,spd)*TimeStep;

CartCordHold(1,x) = pol2cart(PolarHdg,PolarInc);

PolarHdg = PolarHdg +pi/2;  % adjust for pol2cart deffic

if PolarHdg > 2*pi
    PolarHdg = PolarHdg - 2*pi;
end
CartCordHold(1,y) = pol2cart(PolarHdg,PolarInc);

ActAtt(12,x) = ActAtt(12,x) + CartCordHold(1,x);
ActAtt(12,y) = ActAtt(12,y) + CartCordHold(1,y);

for a1=1:NumAct
    for a2=1:NumAct
        ActDist(a1,a2) = sqrt((ActAtt(a2,1) - ActAtt(a1,1))^2 +
(ActAtt(a2,2) - ActAtt(a1,2))^2);
    end
end

for row=1:11
    for col=1:11
        if row > col  % downlink
            if row == 11 % BS TX to SS

                % Power of Signal for downlink (will occupy the bottom
portion of matrix)
                SsRxPowLin =
(BsTxPowLin*BsGainLin*SsGainLin*wavelength^2)/(((4*pi)^2)*((ActDist(row
,col)*1000)^2));
```

```
                % Power of Noise for downlink (will occupy the top
portion of matrix)
                SsSnrLin = SsRxPowLin/SsNoPowLin;
                SsSnrDb = pow2db(SsSnrLin);
                SnrMat(row,col) = SsSnrDb;

            else % SS TX to SS

                % Power of Signal for uplink (will occupy the top
portion of matrix)
                SsRxPowLin =
(SsTxPowLin*SsGainLin*SsGainLin*wavelength^2)/(((4*pi)^2)*((ActDist(row
,col)*1000)^2));

                % Power of Noise for uplink (will occupy the bottom
portion of matrix)
                SsSnrLin = SsRxPowLin/SsNoPowLin;
                SsSnrDb = pow2db(SsSnrLin);
                SnrMat(row,col) = SsSnrDb;

            end

        elseif row < col % uplink

            if col == 11 % SS TX to BS

                % Power of Signal for uplink (will occupy the top
portion of matrix)
                BsRxPowLin =
(SsTxPowLin*SsGainLin*BsGainLin*wavelength^2)/(((4*pi)^2)*((ActDist(row
,col)*1000)^2));

                % Power of Noise for uplink (will occupy the bottom
portion of matrix)
                BsSnrLin = BsRxPowLin/BsNoPowLin;
                BsSnrDb = pow2db(BsSnrLin);
                SnrMat(row,col) = BsSnrDb;

            else % SS TX to SS

                % Power of Signal for uplink (will occupy the top
portion of matrix)
                SsRxPowLin =
(SsTxPowLin*SsGainLin*SsGainLin*wavelength^2)/(((4*pi)^2)*((ActDist(row
,col)*1000)^2));

                % Power of Noise for uplink (will occupy the bottom
portion of matrix)
                 SsSnrLin = SsRxPowLin/SsNoPowLin;
                SsSnrDb = pow2db(SsSnrLin);
                SnrMat(row,col) = SsSnrDb;
            end

        else % do nothing because does not TX to itself
            testz = 5;
```

```
        end

    end
end

% plots positions

scatter(ActAtt(:,x),ActAtt(:,y));
axis([-15 15 -15 15])

% mod for movement demonstration below

ActAtt(1,1) = ActAtt(1,1) - 0.00004947;
ActAtt(5,1) = ActAtt(5,1) - 0.00010289;
ActAtt(6,1) = ActAtt(6,1) - 0.00010289;

ActAttOut = ActAtt;
```

B. BS SYSTEM

1. BS MAC Reciever

```
function [VehDataOut, MsgChanVehMat, MsgChanVehMatT2, MsgChanVehLen]  =
BSmacRX(VehDataIn, ChanVehMat, ChanRawMat, VehCh1, VehCh2, VehCh3,
VehCh4, VehCh5, VehCh6, VehCh7, VehCh8)  % rate of transmission and
transmission frame size

eml.extrinsic('bi2de', 'xor');

NumChan = 4;
NumVeh = 6;

ULalloLen = 28512;  % 28248;
ULsymb = 6336;
DLalloLen = 3456; % 3424;
DLsymb = 768;
MAPlen = 384;
MAPsymb = 768;


VehDataOut = VehDataIn;

% assume rateID to be static among chanel and for vehicle

FrLen = 0;
VehID = 0;

MsgChanVehMat = zeros(ULalloLen,4,NumVeh);
MsgChanVehMatT1 = zeros(ULalloLen,4,NumVeh);
MsgChanVehMatT2 = zeros(ULalloLen,4,NumVeh);
```

```
MsgChanVehLen = zeros(1,4,NumVeh);
MsgChanVehLenT = zeros(1,4,NumVeh);

ChanRawMatTcv = zeros(ULalloLen, NumChan, NumVeh);

VehCh = zeros(ULalloLen, 2*NumChan);

VehCh(:,1) = VehCh1;
VehCh(:,2) = VehCh2;
VehCh(:,3) = VehCh3;
VehCh(:,4) = VehCh4;
VehCh(:,5) = VehCh5;
VehCh(:,6) = VehCh6;
VehCh(:,7) = VehCh7;
VehCh(:,8) = VehCh8;

x =0;
for c = 1:NumChan
    for v = 1:NumVeh
        if (ChanVehMat(v,c) > 0)
            x = x+1;
            ChanRawMatTcv(:,c,v) = VehCh(:,x);
        end
    end
end

g = 0;

FrCount = zeros(NumVeh,1);
FrLoss = zeros(NumVeh,1);
throughput = zeros(NumVeh,1);
biterror = zeros(NumVeh,1);

ChanRawMatTz = zeros(ULalloLen,1);
FrCountData = 5;
FrLossData = 6;
throughputData = 2;
biterrorData = 3;

idx = 0;

for c=1:NumChan
    idx = 0;
    for v=1:NumVeh

        if ChanVehMat(v,c) > 0  % using SS derived for now until have
the matchup right (has 48 less than BS info)
            g =
sum(xor(ChanRawMat(idx+1:idx+48,c),ChanRawMatTcv(idx+1:idx+48,c,v)));
% error checking
            FrCount(v,1) = FrCount(v,1) + 1;
```

```
            FrLen = bi2de(ChanRawMat(idx+9:idx+24,c)','left-msb');  %
read length
            FrLen = FrLen*8;  % change to bits
            VehID = bi2de(ChanRawMat(idx+25:idx+40,c)','left-msb');
% read cid
            VehID = VehID - 65280;  % convert from code
            if (FrLen <= ULalloLen)
                if ((FrLen == (ChanVehMat(v,c))) && VehID == v)
% corroborate cid; corroborate lenght
                    MsgChanVehMat(1:(FrLen-48),c,v) =
ChanRawMat(idx+49:idx+FrLen,c);             % write to Matrix without
header
                    MsgChanVehMatT1(1:(FrLen-48),c,v) =
ChanRawMatTcv(idx+49:idx+FrLen,c,v);       %%%%%%%%%%       % for
throughput and error detection
                    MsgChanVehLenT(1,c,v) = FrLen;  %%%%%%%%%%       %
for throughput
                    MsgChanVehLen(1,c,v) = FrLen - 48;  % shorten to
without header allotment
                end
            end

            if g >= 1 && (FrLen <= ULalloLen)  % upper limit only for
MATLAB considerations
                % counts number of errors in header and zeros out frame
if too many errors for packet loss

                FrLoss(v,1) = FrLoss(v,1) + 1;
                MsgChanVehMatT2(1:(FrLen-48),c,v) =
ChanRawMatTz(idx+49:idx+FrLen,1);
            elseif (FrLen <= ULalloLen)
                MsgChanVehMatT2(1:(FrLen-48),c,v) =
ChanRawMatTcv(idx+49:idx+FrLen,c,v);  % otherwise just writes
transmitted info
            end
            idx = idx + (ChanVehMat(v,c));
        end
    end
end

for v=1:NumVeh

    biterror(v,1) =
sum(sum(xor(MsgChanVehMatT1(:,:,v),MsgChanVehMat(:,:,v))));
    throughput(v,1) = sum(MsgChanVehLenT(1,:,v));

    VehDataOut(v,FrCountData) = FrCount(v,1);
    VehDataOut(v,throughputData) =  throughput(v,1);
    VehDataOut(v,biterrorData) =  biterror(v,1);
    VehDataOut(v,FrLossData) = FrLoss(v,1);
end
```

2. BS MAC Assembley

```
function [VehDataOut, MsgRemMatOut, MsgRemMatOutT,  MsgRemLenOut,
SSreq, MsgSeqOut, RelMsgMat, RelMsgNum ]  = BSmacASSY(VehDataIn,
VehSnrDataIn, Time, MsgChanVehMat, MsgChanVehMatT, MsgChanVehLen,
MsgRemMatIn, MsgRemMatInT, MsgRemLenIn, SSreqOld, MsgSeqIn)  % rate of
transmission and transmission frame size

eml.extrinsic('bi2de', 'xor', 'de2bi');

VehSnrDataOut =VehSnrDataIn;

ULalloLen = 28512;
ULsymb = 6336;
DLalloLen = 3456;
DLsymb = 768;
MAPlen = 384;
MAPsymb = 768;

MaxMsgSize = 524280; % replace with max vid size once we get there

MaxMsgRemSize = 1000;

NumVeh = 6;

NumChan = 4;

RelMsgMat = zeros(5,NumVeh,64,2);
RelMsgNum = zeros(5,NumVeh);

MsgRemMatOut = zeros(MaxMsgRemSize,NumVeh);
MsgRemMatOutT = zeros(MaxMsgRemSize,NumVeh);
MsgRemLenOut = zeros(1,NumVeh);

MsgRemMat = zeros(MaxMsgRemSize,NumVeh);
MsgRemMatT = zeros(MaxMsgRemSize,NumVeh);
MsgRemLen = zeros(1,NumVeh);

MsgRemMat = MsgRemMatIn;
MsgRemMatT = MsgRemMatInT;  % edited
MsgRemLen = MsgRemLenIn;

ContUNum = zeros(NumVeh,1);
RadNum = zeros(NumVeh,1);
VidNum = zeros(NumVeh,1);
ContUByte = zeros(NumVeh,1);
RadByte = zeros(NumVeh,1);
VidByte = zeros(NumVeh,1);

StatUCount = zeros(NumVeh,1);
ContUCount = zeros(NumVeh,1);
RadCount = zeros(NumVeh,1);
VidCount = zeros(NumVeh,1);
```

```
StatUCountB = zeros(NumVeh,1);
ContUCountB = zeros(NumVeh,1);
RadCountB = zeros(NumVeh,1);
VidCountB = zeros(NumVeh,1);

MsgSeqOut = MsgSeqIn;

SSreq = zeros(NumVeh,7);

MsgHdr = zeros(16,1);
MsgBuff = zeros(ULalloLen*NumChan+MaxMsgSize*2,1);  % needs to be
ULalloLen + Maximum Video Message Size
MsgBuffT = zeros(ULalloLen*NumChan+MaxMsgSize*2,1);
MsgBuffLen = length(MsgBuff);

StatULen = 144;
ContULen = 128;

MsgVeh = 0;
MsgType = 0;
MsgLen = 0;
MsgSeq = 0;
MsgTime = 0;

i = 0; % last bit of message

i2 = 0; % progress in reading message
z = 0;
testz = 0;
PacketTypes = 5;

g = 0;
goodput = zeros(NumVeh,1);
packetloss = zeros(NumVeh,PacketTypes); %
packetdelay = zeros(NumVeh,PacketTypes);
packetcount = zeros(NumVeh,PacketTypes);
maxpacketdelay = zeros(NumVeh,PacketTypes);

VehSnrData = 1;
GoodputData = 4;
PkCountStatData = 7;
PkCountContUData = 8;
PkCountRadarData = 9;
PkCountVidData = 10;
PkLossStatData = 11;
PkLossContUData = 12;
PkLossRadarData = 13;
PkLossVidData = 14;
PkAvgDelayStatData = 19;
PkAvgDelayContUData = 20;
PkAvgDelayRadarData = 21;
PkAvgDelayVidData = 22;
PkMaxDelayStatData = 23;
PkMaxDelayContUData = 24;
```

```
PkMaxDelayRadarData = 25;
PkMaxDelayVidData = 26;

VehDataOut = zeros(NumVeh,26);

VehDataOut = VehDataIn;

n2=0;
n3=0;
n4=0;
n5=0;

for v=1:NumVeh

    n2=0;
    n3=0;
    n4=0;
    n5=0;

    i = 0;
    i2 = 0;
    if MsgRemLen(1,v) > 0
        % for mod if MsgRemLen > 1000 then leave the rest as zeros from
MsgBuff and do same for MsgBuffT -- done
        if MsgRemLen(1,v) < MaxMsgRemSize
            MsgBuff(1+i:MsgRemLen(1,v)+i,1) =
MsgRemMat(1:MsgRemLen(1,v),v);
            MsgBuffT(1+i:MsgRemLen(1,v)+i,1) =
MsgRemMatT(1:MsgRemLen(1,v),v);

        else

            MsgBuff(1+i:MaxMsgRemSize+i,1) =
MsgRemMat(1:MaxMsgRemSize,v);
            MsgBuffT(1+i:MaxMsgRemSize+i,1) =
MsgRemMatT(1:MaxMsgRemSize,v);
        end

        i = i + MsgRemLen(1,v);
    end

    for c=1:NumChan
        z = MsgChanVehLen(1,c,v);
        if z > 0 && z <= ULalloLen
            MsgBuff(1+i:z+i,1) = MsgChanVehMat(1:z,c,v);  % create full
length msg
            MsgBuffT(1+i:z+i,1) = MsgChanVehMatT(1:z,c,v);
            i = i + z;
        end
        % i symbolizes the last bit (length), and the first bit should
be the start of a header
    end
    % i2 is progress of reading
```

```
    MaxNum = ceil(MsgBuffLen/128);

    for x = 1:MaxNum
        if (i-i2 >= 48) % if at least header sent it is a message
            MsgVeh = bi2de(MsgBuff(i2+1:i2+4,1)','left-msb');
            MsgType = bi2de(MsgBuff(i2+5:i2+8,1)','left-msb');
            MsgLen = bi2de(MsgBuff(i2+9:i2+24,1)','left-msb');
            MsgSeq = bi2de(MsgBuff(i2+25:i2+32,1)','left-msb');
            MsgTime = bi2de(MsgBuff(i2+33:i2+48,1)','left-msb');
            MsgLen = MsgLen*8;
            g = sum(xor(MsgBuffT(i2+1:i2+48,1),MsgBuff(i2+1:i2+48,1)));

            if (i-i2 >= MsgLen && MsgLen >0 && MsgType > 0) % if
complete message, break down message and record new i2
                if v == 6
                    testz = 5;
                end
                if(MsgType == 2)  %%%%%%%%%%%%%%%%%%%%%%%% working here
                    StatUCount(v,1) = StatUCount(v,1)+1;
                    if (MsgSeq < 257 && (MsgSeq ==
(MsgSeqIn(v,2)+StatUCount(v,1)) || MsgSeq ==
(MsgSeqIn(v,2)+StatUCount(v,1))-256) )
                        MsgSeqOut(v,2) = MsgSeq;
                    elseif (MsgSeq == 1 &&
(MsgSeqIn(v,2)+StatUCount(v,1)) == 257)
                        MsgSeqOut(v,2) = MsgSeq;
                    else
                        MsgSeqOut(v,2) = MsgSeq;
                    end

                    if g >= 1 || n2>63 % error checking
                        packetloss(MsgVeh,MsgType) =
packetloss(MsgVeh,MsgType) + 1; %%change
                    else
                        n2 = n2+1;  % always make available for send to
next hop
                        RelMsgMat(MsgType,v,n2,1) = MsgTime;
                        RelMsgMat(MsgType,v,n2,2) = MsgLen;
                        ContUNum(MsgVeh,1) =
bi2de(MsgBuff(i2+57:i2+64,1)','left-msb');
                        ContUByte(MsgVeh,1) =
bi2de(MsgBuff(i2+68:i2+80,1)','left-msb');
                        RadNum(MsgVeh,1) =
bi2de(MsgBuff(i2+89:i2+96,1)','left-msb');
                        RadByte(MsgVeh,1) =
bi2de(MsgBuff(i2+97:i2+112,1)','left-msb');
                        VidNum(MsgVeh,1) =
bi2de(MsgBuff(i2+121:i2+128,1)','left-msb');
                        VidByte(MsgVeh,1) =
bi2de(MsgBuff(i2+129:i2+144,1)','left-msb');
                        goodput(v,1) = goodput(v,1) + MsgLen - 48;

                    end
```

```
                    packetcount(MsgVeh,MsgType) =
packetcount(MsgVeh,MsgType) + 1; %%change
                    packetdelay(MsgVeh,MsgType) =
packetdelay(MsgVeh,MsgType) + mod(Time-MsgTime,65536); %%change
                    if mod(Time-MsgTime,65536) >
maxpacketdelay(MsgVeh,MsgType)
                        maxpacketdelay(MsgVeh,MsgType) = mod(Time-
MsgTime,65536);
                    end

                elseif(MsgType == 3)
                    ContUCount(v,1) =  ContUCount(v,1) + 1;
                    ContUCountB(MsgVeh,1) =  ContUCountB(MsgVeh,1) +
MsgLen; %%change
                    if (MsgSeq < 257 && (MsgSeq ==
(MsgSeqIn(v,3)+ContUCount(v,1)) || MsgSeq ==
(MsgSeqIn(v,3)+ContUCount(v,1))-256) )
                        MsgSeqOut(v,3) = MsgSeq;
                    elseif (MsgSeq == 1 &&
(MsgSeqIn(v,3)+ContUCount(v,1)) == 257)
                        MsgSeqOut(v,3) = MsgSeq;
                    else
                        MsgSeqOut(v,3) = MsgSeq;
                    end

                    if g >= 1 || n3>63 % error checking
                        packetloss(MsgVeh,MsgType) =
packetloss(MsgVeh,MsgType) + 1; %%change

                    else % send it to next hop
                        n3 = n3+1;
                        RelMsgMat(MsgType,v,n3,1) = MsgTime;
                        RelMsgMat(MsgType,v,n3,2) = MsgLen;
                        goodput(v,1) = goodput(v,1) + MsgLen - 48;

                    end
                    packetcount(MsgVeh,MsgType) =
packetcount(MsgVeh,MsgType) + 1; %%change
                    packetdelay(MsgVeh,MsgType) =
packetdelay(MsgVeh,MsgType) + mod(Time-MsgTime,65536); %%change
                    if mod(Time-MsgTime,65536) >
maxpacketdelay(MsgVeh,MsgType)
                        maxpacketdelay(MsgVeh,MsgType) = mod(Time-
MsgTime,65536);
                    end

                elseif(MsgType == 4)
                    RadCount(v,1) =  RadCount(v,1) + 1;
                    RadCountB(MsgVeh,1) =  RadCountB(MsgVeh,1) +
MsgLen; %%change
                    if (MsgSeq < 257 && (MsgSeq ==
(MsgSeqIn(v,4)+RadCount(v,1))  || MsgSeq ==
(MsgSeqIn(v,4)+RadCount(v,1))-256))
                        MsgSeqOut(v,4) = MsgSeq;
```

```
                    elseif (MsgSeq == 1 &&
(MsgSeqIn(v,4)+RadCount(v,1)) == 257)
                        MsgSeqOut(v,4) = MsgSeq;
                    else
                        MsgSeqOut(v,4) = MsgSeq;
                    end

                    if g >= 1 || n4>63 % error checking
                        packetloss(MsgVeh,MsgType) =
packetloss(MsgVeh,MsgType) + 1; %%change

                    else % send it to next hop
                        n4 = n4+1;
                        RelMsgMat(MsgType,v,n4,1) = MsgTime;
                        RelMsgMat(MsgType,v,n4,2) = MsgLen;
                        goodput(v,1) = goodput(v,1) + MsgLen - 48;
                    end
                    packetcount(MsgVeh,MsgType) =
packetcount(MsgVeh,MsgType) + 1; %%change
                    packetdelay(MsgVeh,MsgType) =
packetdelay(MsgVeh,MsgType) + mod(Time-MsgTime,65536); %%change
                    if mod(Time-MsgTime,65536) >
maxpacketdelay(MsgVeh,MsgType)
                        maxpacketdelay(MsgVeh,MsgType) = mod(Time-
MsgTime,65536);
                    end

                elseif(MsgType == 5)
                    VidCount(v,1) =  VidCount(v,1) + 1;
                    VidCountB(MsgVeh,1) =  VidCountB(MsgVeh,1) +
MsgLen; %%change
                    if (MsgSeq < 257 && (MsgSeq ==
(MsgSeqIn(v,5)+VidCount(v,1)) || MsgSeq ==
(MsgSeqIn(v,5)+VidCount(v,1))-256))
                        MsgSeqOut(v,5) = MsgSeq;
                    elseif (MsgSeq == 1 &&
(MsgSeqIn(v,5)+VidCount(v,1)) == 257)
                        MsgSeqOut(v,5) = MsgSeq;
                    else
                        MsgSeqOut(v,5) = MsgSeq;
                    end

                    if g >= 1 || n5>63 % error checking
                        packetloss(MsgVeh,MsgType) =
packetloss(MsgVeh,MsgType) + 1; %%change
                    else % send it to next hop
                        n5 = n5+1;
                        RelMsgMat(MsgType,v,n5,1) = MsgTime;
                        RelMsgMat(MsgType,v,n5,2) = MsgLen;
                        goodput(v,1) = goodput(v,1) + MsgLen - 48;
                    end
                    packetcount(MsgVeh,MsgType) =
packetcount(MsgVeh,MsgType) + 1; %%change
                    packetdelay(MsgVeh,MsgType) =
packetdelay(MsgVeh,MsgType) + mod(Time-MsgTime,65536); %%change
```

```
                    if mod(Time-MsgTime,65536) >
maxpacketdelay(MsgVeh,MsgType)
                        maxpacketdelay(MsgVeh,MsgType) = mod(Time-
MsgTime,65536);
                    end
                elseif(MsgType == 9)
                else
                    testz = 5;
                end
                i2 = i2 + MsgLen;  % i2 is progress in reading message
            else  % if incomplete message%write to MsgRem
                % for mod if i-i2 > 1000 then only write first 1000 to
BSMsgRemMat and do same for error checking
                % if i-i2 <= 1000 below code works fine

                if i-i2 < MaxMsgRemSize
                    MsgRemMatOut(1:(i-i2),v) = MsgBuff(i2+1:i,1);
                    MsgRemMatOutT(1:(i-i2),v) = MsgBuffT(i2+1:i,1);  %
error checking

                else
                    MsgRemMatOut(1:MaxMsgRemSize,v) =
MsgBuff(i2+1:i2+1000,1);
                    MsgRemMatOutT(1:MaxMsgRemSize,v) =
MsgBuffT(i2+1:i2+1000,1);  % error checking
                end
                MsgRemLenOut(1,v) = i-i2;
                i2 = i; % new
            end

        elseif(i-i2 > 0)  % if incomplete message%write to MsgRem
            % for mod if i-i2 > 1000 then only write first 1000 to
BSMsgRemMat and do same for error checking
            % if i-i2 <= 1000 below code works fine

            if i-i2 < MaxMsgRemSize
                MsgRemMatOut(1:(i-i2),v) = MsgBuff(i2+1:i,1);
                MsgRemMatOutT(1:(i-i2),v) = MsgBuffT(i2+1:i,1);  %
error checking
            else
                MsgRemMatOut(1:MaxMsgRemSize,v) =
MsgBuff(i2+1:i2+1000,1);
                MsgRemMatOutT(1:MaxMsgRemSize,v) =
MsgBuffT(i2+1:i2+1000,1);  % error checking
            end

            MsgRemLenOut(1,v) = i-i2;
            i2 = i; % new
        end
    end
    RelMsgNum(2,v) = n2;
    RelMsgNum(3,v) = n3;
    RelMsgNum(4,v) = n4;
    RelMsgNum(5,v) = n5;
```

```
end

for v=1:NumVeh
    SSreq(v,1) = v;  % labels front

    if ContUByte(v,1) > 0
        SSreq(v,5) =  ContUByte(v,1);
    else
        SSreq(v,5) =  SSreqOld(v,5);
    end
    if RadByte(v,1) > 0
        SSreq(v,6) =  RadByte(v,1);
    else
        SSreq(v,6) =  SSreqOld(v,6);
    end

    if VidByte(v,1) > 0
        SSreq(v,7) =  VidByte(v,1);
    else
        SSreq(v,7) =  SSreqOld(v,7);
    end

    if sum(MsgChanVehLen(1,:,v)) > 0 && packetcount(v,2) == 0
        testz = sum(MsgChanVehLen(1,:,v));
        SSreq(v,4:7) = SSreq(v,4:7)*2;
        testz = 0;
    else
        SSreq(v,4) = 32;  % always want to transmit status with other
messages
    end

    VehDataOut(v,VehSnrData) = VehSnrDataOut(v,1);
    VehDataOut(v,GoodputData) = goodput(v,1);
    VehDataOut(v,PkCountStatData) = packetcount(v,2);
    VehDataOut(v,PkCountContUData) = packetcount(v,3);
    VehDataOut(v,PkCountRadarData) =  packetcount(v,4);
    VehDataOut(v,PkCountVidData) =  packetcount(v,5);
    VehDataOut(v,PkLossStatData) =  packetloss(v,2);
    VehDataOut(v,PkLossContUData) =  packetloss(v,3);
    VehDataOut(v,PkLossRadarData) =  packetloss(v,4);
    VehDataOut(v,PkLossVidData) =  packetloss(v,5);
    VehDataOut(v,PkAvgDelayStatData) = packetdelay(v,2);
    VehDataOut(v,PkAvgDelayContUData) = packetdelay(v,3);
    VehDataOut(v,PkAvgDelayRadarData) =  packetdelay(v,4);
    VehDataOut(v,PkAvgDelayVidData) =  packetdelay(v,5);
    VehDataOut(v,PkMaxDelayStatData) =  maxpacketdelay(v,2);
    VehDataOut(v,PkMaxDelayContUData) =  maxpacketdelay(v,3);
    VehDataOut(v,PkMaxDelayRadarData) =  maxpacketdelay(v,4);
    VehDataOut(v,PkMaxDelayVidData) =  maxpacketdelay(v,5);

end

SSreq = abs(SSreq);
```

3. BS MAC UL Controller

```
function [ChanRateIDU, ChanVehMatSnr, ChanVehMat, ChanVehMatBit,
PriNextOut, VehSnrDataOut]  = BSmacCTRLul(UAVatt, Time, RelayEn,
SmallBwEn, SnrMat, SSreq, PriNextIn)  % rate of transmission and
transmission frame size

NumVeh=6;
NumChan = 4;

VehNum = 1;
VehPri = 2;
VehContD = 3;
VehStatU = 4;
VehContU = 5;
VehRadU = 6;
VehVidU = 7;
VehMsgDem = zeros(NumVeh,7);  % information input from SS MACs
ULalloLen = 28512;
ULsymb = 6336;  %ULsymb = 2048;
DLalloLen = 3456;
DLsymb = 768;
MAPlen = 384;
MAPsymb = 768;

SymBitMat = [.5 1 1.5 2 3 4 4.5];

VehSnrDataOut = zeros(NumVeh,1);
VehSnrData = 1;
%%%%%%%%%%%%%%%%%% below has been moded

SnrThresh = [4 10 12 19 22 28];

VehChanRateID = [10 10 10 10;10 10 10 10;10 10 10 10;10 10 10 10;10 10
10 10;10 10 10 10];
VehRateID = zeros(NumVeh,1);

for v=1:NumVeh
    RelayStation = 11;
    for t=1:6
        if UAVatt(v,7) > 0
            RelayStation = UAVatt(v,7);
        end


        if SnrMat(v,RelayStation) > SnrThresh(1,t)

            VehRateID(v,1) = t;
        end
    end
```

```
    VehSnrDataOut(v,VehSnrData) = SnrMat(v,RelayStation);
end

VehMsgDem(1,1:7) = SSreq(1,1:7);
VehMsgDem(2,1:7) = SSreq(2,1:7);
VehMsgDem(3,1:7) = SSreq(3,1:7);
VehMsgDem(4,1:7) = SSreq(4,1:7);
VehMsgDem(5,1:7) = SSreq(5,1:7);
VehMsgDem(6,1:7) = SSreq(6,1:7);

% divide below by symbitmat

VehMsgDem(1,4:7) =
ceil(SSreq(1,4:7)/(2*(SymBitMat(1,min(VehRateID(:,1))+1)))); % stop gap
for now
VehMsgDem(2,4:7) =
ceil(SSreq(2,4:7)/(2*(SymBitMat(1,min(VehRateID(:,1))+1)))); % SSreq is
now in bytes
VehMsgDem(3,4:7) =
ceil(SSreq(3,4:7)/(2*(SymBitMat(1,min(VehRateID(:,1))+1))));
VehMsgDem(4,4:7) =
ceil(SSreq(4,4:7)/(2*(SymBitMat(1,min(VehRateID(:,1))+1))));
VehMsgDem(5,4:7) =
ceil(SSreq(5,4:7)/(2*(SymBitMat(1,min(VehRateID(:,1))+1))));
VehMsgDem(6,4:7) =
ceil(SSreq(6,4:7)/(2*(SymBitMat(1,min(VehRateID(:,1))+1))));

if VehMsgDem(1,1) == 0
    VehMsgDem(1,:) = [1 1 0 20 20 0 0];  % now in symbol chunk denom 1
= 16 symb
end

if VehMsgDem(2,1) == 0
    VehMsgDem(2,:) = [2 2 0 20 20 0 0];
end

if VehMsgDem(3,1) == 0
    VehMsgDem(3,:) = [3 3 0 20 20 0 0];
end

if VehMsgDem(4,1) == 0
    VehMsgDem(4,:) = [4 4 0 20 20 0 0];
end

if VehMsgDem(5,1) == 0
    VehMsgDem(5,:) = [5 5 0 20 20 0 0];
end

if VehMsgDem(6,1) == 0
    VehMsgDem(6,:) = [6 6 0 20 20 0 0];
end

%%%%%%%%%%%%%%%%%%%%%%%%%
```

```
if RelayEn == 1

VehMsgDem(1:NumVeh,VehPri) = [2;3;4;5;6;7];

else

VehMsgDem(1:NumVeh,VehPri) = [7;6;5;4;3;2];

end

PriNextOut = 0;

PriNextOut = PriNextIn;

if (PriNextOut == 0 || PriNextOut > NumVeh)  % constant rotate #1 Pri
    PriNextOut = 1;
end

VehMsgDem(PriNextOut,VehPri) = 1;

VehMsgDem(PriNextOut,VehStatU) = VehMsgDem(PriNextOut,VehStatU) + 8;  %
allow it to transmit status

PriNextOut = PriNextOut + 1;

if mod(Time,4) == 2 && RelayEn == 1

    VehMsgDem(1,:) = [1 7 0 0 0 0 0];
    VehMsgDem(2,:) = [2 6 0 0 0 0 0];
    VehMsgDem(3,:) = [3 5 0 0 0 0 0];
    VehMsgDem(4,:) = [4 4 0 0 0 0 0];
    VehMsgDem(5,2) = 3;
    VehMsgDem(6,2) = 2;
    VehMsgDem(6,4) = VehMsgDem(6,4)+8;

   PriNextOut = PriNextOut - 1;
   if PriNextOut > 4
       PriNextOut = 1;
       if mod(Time,8) == 2
       VehMsgDem(5,2) = 1;
       VehMsgDem(5,4) = VehMsgDem(5,4)+8;
       end
   end

elseif mod(Time,4) == 0 && RelayEn == 1

    VehMsgDem(1,:) = [1 7 0 0 0 0 0];
    VehMsgDem(2,:) = [2 6 0 0 0 0 0];
    VehMsgDem(3,:) = [3 5 0 0 0 0 0];
    VehMsgDem(4,:) = [4 4 0 0 0 0 0];
    VehMsgDem(5,2) = 3;
    VehMsgDem(6,2) = 2;
```

```
   PriNextOut = PriNextOut - 1;
   if PriNextOut > 4
       PriNextOut = 1;
       if mod(Time,8) == 0
       VehMsgDem(5,2) = 1;
       VehMsgDem(5,4) = VehMsgDem(5,4)+8;
       end

   end

elseif RelayEn == 1

    VehMsgDem(5,:) = [5 9 0 0 0 0 0];
    VehMsgDem(6,:) = [6 8 0 0 0 0 0];

end

%sort by priority
VehMsgDem = sortrows(VehMsgDem,VehPri);

SymReqMat = ones(NumVeh,7); % symbols to correspond to vehicle
SNR/burst profile

ChanAllMatU = [ULsymb/16;ULsymb/16;ULsymb/16;ULsymb/16]; % scaled down
for now to represent 1024 bits

%Create copy demand matrix
VehDataDem = VehMsgDem;

%multiply it by Msg SMat

%Multiply it by Symbol Req matrix

SymReqMat = SymReqMat.*VehDataDem;

SysReqMatBackup = SymReqMat;

%Create Available Symb Matrix and Summ it up

ChanAllU = sum(ChanAllMatU); % since no symbols, using data

%Account for Symbol Req in order of priority and zero out exceedances
of Available

%and create channel matrix

ChanVehMat = zeros(NumVeh,NumChan);

% revamp pri matrix for simplicity
```

```
CurrChan = 1;
% protion below is wrong in that it does not have overall
accountability for allocation % changed from 6 to 8 to account for
convolution bits plus header
for row=1:NumVeh % row is vehicle
   for col=4:7 % column is message

       if(SymReqMat(row,col) > 0)

           if (SymReqMat(row,col) + 8 + sum(ChanVehMat(:,CurrChan)) <
ChanAllMatU(CurrChan)) % if plenty of room ChanVehMat(row,CurrChan)
                if(ChanVehMat(row,CurrChan) == 0) % tacks on header
                     ChanVehMat(row,CurrChan) = 8;
                end
                ChanVehMat(row,CurrChan) = SymReqMat(row,col) +
ChanVehMat(row,CurrChan);
                SymReqMat(row,col) = 0;
                if(sum(ChanVehMat(:,CurrChan)) >=
(ChanAllMatU(CurrChan)-8) && sum(ChanVehMat(:,CurrChan)) <
(ChanAllMatU(CurrChan)))  % check if room for another write

ChanVehMat(row,CurrChan)=ChanVehMat(row,CurrChan)+(ChanAllMatU(CurrChan
) - sum(ChanVehMat(:,CurrChan)));
                    CurrChan = CurrChan+1;   %go to next channel
                      if( CurrChan > NumChan )
                           SymReqLeftover = SymReqMat;
                           SymReqMat = zeros(NumVeh, 7);
                           CurrChan = CurrChan - 1;
                      end
                end

           else  % if divide over multiple
                numHead = ceil((SymReqMat(row,col) +16+
sum(ChanVehMat(:,CurrChan)))/(ChanAllMatU(CurrChan))); % calculates
number headers needed
                numdivide = floor((SymReqMat(row,col) + numHead*8 +
sum(ChanVehMat(:,CurrChan)))/(ChanAllMatU(CurrChan)));
%ChanVehMat(row,CurrChan)
                for bin = 1:numdivide
                       if(ChanVehMat(row,CurrChan) == 0)
                           ChanVehMat(row,CurrChan) = 8;
                       end
                       segmentsize = ChanAllMatU(CurrChan)-
sum(ChanVehMat(:,CurrChan)); % segment size  ChanVehMat(row,CurrChan)
                       ChanVehMat(row,CurrChan) =
ChanVehMat(row,CurrChan) +  segmentsize;  % put in segment
                       SymReqMat(row,col) = SymReqMat(row,col) -
segmentsize;  %account for segment
                       if(sum(ChanVehMat(:,CurrChan)) >=
(ChanAllMatU(CurrChan)-8) && sum(ChanVehMat(:,CurrChan)) <
(ChanAllMatU(CurrChan)))  % check if room for another write

ChanVehMat(row,CurrChan)=ChanVehMat(row,CurrChan)+(ChanAllMatU(CurrChan
) - sum(ChanVehMat(:,CurrChan)));
                       end
```

```
                    CurrChan = CurrChan+1;   %go to next channel
                    if( CurrChan > NumChan )
                         SymReqLeftover = SymReqMat;
                         SymReqMat = zeros(NumVeh, 7);
                         CurrChan = CurrChan - 1;
                    end

               end
               if (SymReqMat(row,col) > 0)  %place last segment in next
chanel
                   if(ChanVehMat(row,CurrChan) == 0)
                       ChanVehMat(row,CurrChan) = 8;
                   end
                   ChanVehMat(row,CurrChan) = ChanVehMat(row,CurrChan)
+ SymReqMat(row,col);
                   SymReqMat(row,col) = 0;
                   if(sum(ChanVehMat(:,CurrChan)) >=
(ChanAllMatU(CurrChan)-8) && sum(ChanVehMat(:,CurrChan)) <
(ChanAllMatU(CurrChan)))  % check if room for another write

ChanVehMat(row,CurrChan)=ChanVehMat(row,CurrChan)+(ChanAllMatU(CurrChan
) - sum(ChanVehMat(:,CurrChan)));
                                        CurrChan = CurrChan+1;   %go to
next channel
                    if( CurrChan > NumChan )
                         SymReqLeftover = SymReqMat;
                         SymReqMat = zeros(NumVeh, 7);
                         CurrChan = CurrChan - 1;
                    end
                   end
               end
           end
       end
   end
end

total = 0;
mark = 0;
count = 0;

%% limits BW to 2 channels if BW limited enabled
if SmallBwEn == 1
   ChanVehMat(:,3) = 0;
   ChanVehMat(:,4) = 0;

end

for c = 1:4  % limits channel allocation to only 2 users and creates
VehChanRateID and also ensures minimum fr size of 8 (*16*rate-8) bits
    count = 0;
    for v = 1:NumVeh

        if ChanVehMat(v,c) > 0 && ChanVehMat(v,c) < 9  % added
            ChanVehMat(v,c) = 0;
```

```
        end

        if ChanVehMat(v,c) > 1
            count = count +1;

        end
        if (count > 2)
                ChanVehMat(v,c) = 0;
        end
    end
end

for c = 1:4  % ensures channel is fully allocated if used at all
    total = 0;
    mark = 0;
    for v = 1:NumVeh
        total = total + ChanVehMat(v,c);
        if ChanVehMat(v,c) > 1
            mark = v;
        end
        if (v == NumVeh && total < (ChanAllMatU(c)) && total ~= 0)
                ChanVehMat(mark,c) = ChanVehMat(mark,c) +
((ChanAllMatU(c))-total);
        end

    end
end

% Make ChanVehMat correspond to SysDem (not reading first row of SysDem
as first Veh)

ChanVehMatCorr = zeros(NumVeh,NumChan);

for v = 1:NumVeh

ChanVehMatCorr(SysReqMatBackup(v,1),1:NumChan)=ChanVehMat(v,1:NumChan);
end

ChanVehMat = ChanVehMatCorr;

ChanVehMat = ChanVehMat*16;  % converts into symbols

ChanVehMatSnr = ChanVehMat;

for c = 1:NumChan
   for v = 1:NumVeh

        if UAVatt(v,7) > 0
            RelayStation = UAVatt(v,7);
        else
            RelayStation = 11;
        end
```

```
        if ChanVehMat(v,c) > 0
            VehChanRateID(v,c) = VehRateID(v,1);

            ChanVehMatSnr(v,c) = SnrMat(v,RelayStation);
        end
   end
end

ChanRateIDU = zeros(1,4);  % contains minimum rateID for all vehicles
in Channel

ChanVehMatBit = zeros(NumVeh, NumChan);

for c = 1:NumChan
   ChanRateIDU(1,c) = min(VehChanRateID(:,c));
   if(ChanRateIDU(1,c) < 10)
   ChanVehMatBit(:,c) =
ChanVehMat(:,c)*SymBitMat(1,ChanRateIDU(1,c)+1);
   else
       ChanRateIDU(1,c) = 0;
   end

end

for c = 1:4  % deletes last 8 bits for convolution from each channel
    for v = 1:NumVeh
        if ChanVehMatBit(v,c) > 0 && v == NumVeh
            ChanVehMatBit(v,c) = ChanVehMatBit(v,c) - 8;
        elseif ChanVehMatBit(v,c) > 0 &&
sum(ChanVehMatBit(v+1:NumVeh,c)) == 0
            ChanVehMatBit(v,c) = ChanVehMatBit(v,c) - 8;
        end
    end
end
```

4. BS MAC DL Controller

```
function [VehDataOut, ChanRateIDD, ChanVehMatSnr, ChanVehMat,VehTX,
PriNextOut, V1MsgCtOut, V1MsgRemOut, MsgAttOut]  =
BSmacCTRLdl(RelMsgMatDL, RelMsgNumDL, UAVatt, RelayEn, Time, SmallBwEn,
SnrMat, inFR1, inFR2, inFR3, inFR4, PriNextIn, V1MsgCtIn, V1MsgRemIn,
MsgAttIn)  % rate of transmission and transmission frame size

eml.extrinsic('de2bi');

NumVeh=6;
NumChan = 4;

VehNum = 1;
VehPri = 2;
VehContD = 3;
VehStatU = 4;
```

```
VehContU = 5;
VehRadU = 6;
VehVidU = 7;

MaxRelayNum = 3;

VehRelaying = zeros(MaxRelayNum,NumVeh);

for v=1:NumVeh
    RelayNum = 0;
    for r=1:MaxRelayNum
        if UAVatt(v,7+r) > 0
            RelayNum = RelayNum+1;
            VehRelaying(RelayNum,v) = UAVatt(v,7+r);
        end
    end
end

ULalloLen = 28512;
ULsymb = 6336;
DLalloLen = 3456;
DLsymb = 768;
MAPlen = 384;
MAPsymb = 768;
ContDLen = 560;
testz=0;

MaxLen = ContDLen;  % add MaxLen of 1000 with MsgTemp that long  %
changed from 900
MsgTemp = zeros(MaxLen+DLalloLen,NumChan);% 29148 for now
MsgTemp(:,1) = inFR1(1:MaxLen+DLalloLen,1);
MsgTemp(:,2) = inFR2(1:MaxLen+DLalloLen,1);
MsgTemp(:,3) = inFR3(1:MaxLen+DLalloLen,1);
MsgTemp(:,4) = inFR4(1:MaxLen+DLalloLen,1);

VehMsgDem = zeros(NumVeh,7);  % information input from SS MACs

VehMsgDem(1,:) = [1 1 0 0 0 0 0];
VehMsgDem(2,:) = [2 2 0 0 0 0 0];
VehMsgDem(3,:) = [3 3 0 0 0 0 0];
VehMsgDem(4,:) = [4 4 0 0 0 0 0];
VehMsgDem(5,:) = [5 5 0 0 0 0 0];
VehMsgDem(6,:) = [6 6 0 0 0 0 0];

CurrTime = mod(V1MsgCtIn(7,1)+1,65536);   % change periodicity of
message of CondDMsg

for v=1:NumVeh
    if (mod(CurrTime+v,7) == 0)
        VehMsgDem(v,3) = 1;
    end
end
```

```
BuffSize = 64;
%%%%%%%%%%%%%%%%%%%%%%%%

%start
MsgAttOut = zeros(BuffSize,4,NumVeh);  % buffer holding message
attributes  %
MsgAttOut = MsgAttIn;

% change to manageable size for bpsk for now
ContDMsg = zeros(ContDLen,1);
ContDMsgNum = VehMsgDem(:,3);
ContDNewMsg = ContDMsgNum;
ContDSeqNum =  V1MsgCtIn(6,:)';

%test
rg = zeros(NumVeh,1);
for v=1:NumVeh
    rg(v,1) = sum(MsgAttOut(:,3,v));
end

packetdrop = zeros(NumVeh,1);

for v=1:NumVeh

    %%%%%%%%%%%%%% Add Message to queue
    ContDMsgCt = sum(MsgAttOut(:,3,v));
    if ContDNewMsg(v,1) + ContDMsgCt > BuffSize
        packetdrop(v,1) = ContDNewMsg(v,1) + ContDMsgCt - BuffSize;
        ContDNewMsg(v,1) = BuffSize - ContDMsgCt;

    end

    if ContDNewMsg(v,1) > 0
        for x=1:ContDNewMsg(v,1)
            i = mod(ContDSeqNum(v,1)-1+ContDMsgCt+x,BuffSize)+1; %
starts at 1; range 1-1024
            %%% enter length
            MsgAttOut(i,1,v) = ContDLen;
            %%% enter time
            MsgAttOut(i,2,v) = CurrTime;
            %%% enter count
            MsgAttOut(i,3,v) = 1;
            %%%% enter vehicle
            MsgAttOut(i,4,v) = v;

        end
    end

    for r=1:MaxRelayNum

        if VehRelaying(r,v) > 0

            ContDNewMsgRel = RelMsgNumDL(1,VehRelaying(r,v));
```

```
            ContDMsgCtRel = sum(MsgAttOut(:,3,v));
            if ContDNewMsgRel + ContDMsgCtRel > BuffSize
                packetdrop(VehRelaying(r,v),1) = ContDNewMsgRel +
ContDMsgCtRel - BuffSize;
                ContDNewMsgRel = BuffSize - ContDMsgCtRel;
            end

            if ContDNewMsgRel > 0
                for x=1:ContDNewMsgRel
                    i = mod(ContDSeqNum(v,1)-
1+x+ContDMsgCtRel,BuffSize)+1; % starts at 1; range 1-1024
%+StatUMsgCt taken out
                    %%% enter length
                    MsgAttOut(i,1,v) = ContDLen;
                    %%% enter time
                    MsgAttOut(i,2,v) =
RelMsgMatDL(1,VehRelaying(r,v),x,1);
                    %%% enter count
                    MsgAttOut(i,3,v) = 1;
                    %%%% enter vehicle
                    MsgAttOut(i,4,v) = VehRelaying(r,v);

                end
            end
        end
    end
end

PkDropContDData = 16;

VehDataOut = zeros(NumVeh,26);

for v=1:NumVeh

    VehDataOut(v,PkDropContDData) = packetdrop(v,1);

end

VehDataDem = zeros(NumVeh,7);
VehDataDem(1,:) = [1 1 0 0 0 0 0];
VehDataDem(2,:) = [2 2 0 0 0 0 0];
VehDataDem(3,:) = [3 3 0 0 0 0 0];
VehDataDem(4,:) = [4 4 0 0 0 0 0];
VehDataDem(5,:) = [5 5 0 0 0 0 0];
VehDataDem(6,:) = [6 6 0 0 0 0 0];

for v = 1:NumVeh
    VehDataDem(v,3) = sum(MsgAttOut(:,1,v)) + V1MsgCtIn(4,v);
end

VehDataDem(:,3) = ceil(VehDataDem(:,3)/8);  % stop gap measure for now

VehDataDem(1:NumVeh,VehPri) = [2;3;4;5;6;7];
```

```
PriNextOut = 0;

PriNextOut = PriNextIn;

if (PriNextOut == 0 || PriNextOut > NumVeh)  % constant rotate #1 Pri
    PriNextOut = 1;
end

VehDataDem(PriNextOut,VehPri) = 1;

PriNextOut = PriNextOut + 1;

RelayStation = 11;

% relay
if mod(Time,4) == 2 && RelayEn == 1

    VehDataDem(1,:) = [1 4 0 0 0 0 0];
    VehDataDem(2,:) = [2 5 0 0 0 0 0];
    VehDataDem(3,:) = [3 6 0 0 0 0 0];
    VehDataDem(4,:) = [4 7 0 0 0 0 0];
    VehDataDem(5,2) = 2;
    VehDataDem(6,2) = 3;

    PriNextOut = PriNextOut - 1;
    if PriNextOut > 4
        PriNextOut = 1;
    end

elseif mod(Time,4) == 0 && RelayEn == 1

    VehDataDem(1,:) = [1 4 0 0 0 0 0];
    VehDataDem(2,:) = [2 5 0 0 0 0 0];
    VehDataDem(3,:) = [3 6 0 0 0 0 0];
    VehDataDem(4,:) = [4 7 0 0 0 0 0];
    VehDataDem(5,2) = 3;
    VehDataDem(6,2) = 2;

    PriNextOut = PriNextOut - 1;
    if PriNextOut > 4
        PriNextOut = 1;
    end

elseif RelayEn == 1

    VehDataDem(5,:) = [5 8 0 0 0 0 0];
    VehDataDem(6,:) = [6 9 0 0 0 0 0];

end

%sort by priority
```

```
VehDataDem = sortrows(VehDataDem,VehPri);

SymReqMat = ones(NumVeh,7); % symbols to correspond to vehicle
SNR/burst profile

ChanAllMatD = [DLsymb/16;DLsymb/16;DLsymb/16;DLsymb/16];

%Create copy demand matrix

%multiply it by Msg Sz Mat

%Multiply it by Symbol Req matrix

SymReqMat = SymReqMat.*VehDataDem;

SysReqMatBackup = SymReqMat;

%Create Available Symb Matrix and Summ it up

%Account for Symbol Req in order of priority and zero out exceedances
of Available

%and create channel matrix

ChanVehMat = zeros(NumVeh,NumChan);

% revamp pri matrix for simplicity

CurrChan = 1;
% protion below is wrong in that it does not have overall
accountability for allocation % changed from 6 to 8 to account for
convolution bits plus header
for row=1:NumVeh % row is vehicle
    col = 3;
    if(SymReqMat(row,col) > 0)

        if (SymReqMat(row,col) + 8 + sum(ChanVehMat(:,CurrChan)) <
ChanAllMatD(CurrChan)) % if plenty of room ChanVehMat(row,CurrChan)
            if(ChanVehMat(row,CurrChan) == 0) % tacks on header
                ChanVehMat(row,CurrChan) = 8;
            end
            ChanVehMat(row,CurrChan) = SymReqMat(row,col) +
ChanVehMat(row,CurrChan);
            SymReqMat(row,col) = 0;
            if(sum(ChanVehMat(:,CurrChan)) >= (ChanAllMatD(CurrChan)-8)
&& sum(ChanVehMat(:,CurrChan)) < (ChanAllMatD(CurrChan)))  % check if
room for another write

ChanVehMat(row,CurrChan)=ChanVehMat(row,CurrChan)+(ChanAllMatD(CurrChan
) - sum(ChanVehMat(:,CurrChan)));
                CurrChan = CurrChan+1;   %go to next channel
                if( CurrChan > NumChan )
                    SymReqLeftover = SymReqMat;
```

```
                    SymReqMat = zeros(NumVeh, 7);
                    CurrChan = CurrChan - 1;
                end
            end

        else  % if divide over multiple
            numHead = ceil((SymReqMat(row,col) +
16+sum(ChanVehMat(:,CurrChan)))/(ChanAllMatD(CurrChan))); % calculates
number headers needed
            numdivide = floor((SymReqMat(row,col) + numHead*8 +
sum(ChanVehMat(:,CurrChan)))/(ChanAllMatD(CurrChan)));
%ChanVehMat(row,CurrChan)
            for bin = 1:numdivide
                if(ChanVehMat(row,CurrChan) == 0)
                    ChanVehMat(row,CurrChan) = 8;
                end
                segmentsize = ChanAllMatD(CurrChan)-
sum(ChanVehMat(:,CurrChan)); % segment size  ChanVehMat(row,CurrChan)
                ChanVehMat(row,CurrChan) = ChanVehMat(row,CurrChan) +
segmentsize;  % put in segment
                SymReqMat(row,col) = SymReqMat(row,col) - segmentsize;
%account for segment
                if(sum(ChanVehMat(:,CurrChan)) >=
(ChanAllMatD(CurrChan)-8) && sum(ChanVehMat(:,CurrChan)) <
(ChanAllMatD(CurrChan)))  % check if room for another write

ChanVehMat(row,CurrChan)=ChanVehMat(row,CurrChan)+(ChanAllMatD(CurrChan
) - sum(ChanVehMat(:,CurrChan)));
                end
                CurrChan = CurrChan+1;   %go to next channel
                if( CurrChan > NumChan )
                    SymReqLeftover = SymReqMat;
                    SymReqMat = zeros(NumVeh, 7);
                    CurrChan = CurrChan - 1;
                end

            end
            if (SymReqMat(row,col) > 0)  %place last segment in next
chanel
                if(ChanVehMat(row,CurrChan) == 0)
                    ChanVehMat(row,CurrChan) = 8;
                end
                ChanVehMat(row,CurrChan) = ChanVehMat(row,CurrChan) +
SymReqMat(row,col);
                SymReqMat(row,col) = 0;
                if(sum(ChanVehMat(:,CurrChan)) >=
(ChanAllMatD(CurrChan)-8) && sum(ChanVehMat(:,CurrChan)) <
(ChanAllMatD(CurrChan)))  % check if room for another write

ChanVehMat(row,CurrChan)=ChanVehMat(row,CurrChan)+(ChanAllMatD(CurrChan
) - sum(ChanVehMat(:,CurrChan)));
                    CurrChan = CurrChan+1;   %go to next channel
                    if( CurrChan > NumChan )
                        SymReqLeftover = SymReqMat;
                        SymReqMat = zeros(NumVeh, 7);
```

```
                        CurrChan = CurrChan - 1;
                    end
                end
            end
        end
    end
end

total = 0;
mark = 0;
count = 0;

%% limits BW to 2 channels if BW limited enabled
if SmallBwEn == 1
    ChanVehMat(:,3) = 0;
    ChanVehMat(:,4) = 0;

end

for c = 1:4  % limits channel allocation to only 2 users and creates
VehChanRateIDand also ensures minimum fr size of 8 (*16*rate-8) bits
    count = 0;
    for v = 1:NumVeh

        if ChanVehMat(v,c) > 0 && ChanVehMat(v,c) < 9  % added
            ChanVehMat(v,c) = 0;
        end

        if ChanVehMat(v,c) > 1
            count = count +1;

        end
        if (count > 2)
            ChanVehMat(v,c) = 0;
        end
    end
end

for c = 1:4  % ensures channel is fully allocated if used at all
    total = 0;
    mark = 0;
    for v = 1:NumVeh
        total = total + ChanVehMat(v,c);
        if ChanVehMat(v,c) > 1
            mark = v;
        end
        if (v == NumVeh && total < ChanAllMatD(c) && total ~= 0)
            ChanVehMat(mark,c) = ChanVehMat(mark,c) + (ChanAllMatD(c)-
total);
        end
    end
end
```

```
% Make ChanVehMat correspond to SysDem (not reading first row of SysDem
as first Veh)

ChanVehMatCorr = zeros(NumVeh,NumChan);

for v = 1:NumVeh

ChanVehMatCorr(SysReqMatBackup(v,1),1:NumChan)=ChanVehMat(v,1:NumChan);
end

ChanVehMat = ChanVehMatCorr;

ChanVehMat = ChanVehMat*16;

ChanVehMatSnr = ChanVehMat;

VehSnrData = 1;

SnrThresh = [4 10 12 19 22 28];

VehChanRateID = [10 10 10 10;10 10 10 10;10 10 10 10;10 10 10 10;10 10
10 10;10 10 10 10];
VehRateID = zeros(NumVeh,1);

for v=1:NumVeh
    for t=1:6

        if UAVatt(v,7) > 0
            RelayStation = UAVatt(v,7);
        end

        if SnrMat(RelayStation,v) > SnrThresh(1,t)
            VehRateID(v,1) = t;
        end
    end
    VehDataOut(v,VehSnrData) = SnrMat(RelayStation,v);
end

for c = 1:NumChan
    for v = 1:NumVeh

        if UAVatt(v,7) > 0
            RelayStation = UAVatt(v,7);
        else
            RelayStation = 11;
        end

        if ChanVehMat(v,c) > 0
            VehChanRateID(v,c) = VehRateID(v,1);
            ChanVehMatSnr(v,c) = SnrMat(RelayStation,v);

        end
```

```
    end
end

%%%%%%%%%%%%%%%%%%%%%%%%%%%% once have ChanVehMat, write messages (or
at least headers) %%%%%%%%%%%%%%%%%%%%%%
SymBitMat = [.5 1 1.5 2 3 4 4.5];

ChanRateIDD = zeros(1,4);

ChanVehMatBit = zeros(NumVeh, NumChan);
test = zeros(NumVeh, NumChan);

for c = 1:NumChan
    ChanRateIDD(1,c) = min(VehChanRateID(:,c));
    if(ChanRateIDD(1,c) < 10)
        ChanVehMatBit(:,c) =
ChanVehMat(:,c)*SymBitMat(1,ChanRateIDD(1,c)+1);
    else
        ChanRateIDD(1,c) = 0;
    end
end

for c = 1:4  % deletes last 8 bits for convolution from each channel
    for v = 1:NumVeh
        if ChanVehMatBit(v,c) > 0 && v == NumVeh
            ChanVehMatBit(v,c) = ChanVehMatBit(v,c) - 8;
        elseif ChanVehMatBit(v,c) > 0 &&
sum(ChanVehMatBit(v+1:NumVeh,c)) == 0
            ChanVehMatBit(v,c) = ChanVehMatBit(v,c) - 8;
        end
    end
end

VehTX=zeros(DLalloLen,NumChan,NumVeh);  % Vehicle TX according to
channel
VehTXst = zeros(1,NumChan, NumVeh);  % start place includes header (yet
to be added)

MsgLen = 0;
MsgVehBi = zeros(4,1);
MsgTypeBi = zeros(4,1);
MsgLenBi = zeros(16,1);
% StatTypeBi = zeros(8,1);
% StatNumBi = zeros(8,1);
% StatSeqNumBi = zeros(8,1);
ContDTypeBi = zeros(8,1);
ContDSeqNumBi = zeros(8,1);
ContDTimeBi = zeros(16,1);
% StatMsgPadBi = zeros(104,1);
ContDMsgPadBi = zeros(512,1);   % come back to
FillTypeBi = zeros(8,1);
FillSeqNumBi = zeros(8,1);
FillTimeBi = zeros(16,1);
% StatMsgPadBi = zeros(104,1);
```

```
FillMsgPadBi = zeros(160,1);
ContDMsgHdr = zeros(48,1);
FillMsgHdr = zeros(48,1);

% ContDTime = zeros(NumVeh,1);
FillLen = 208;
FillTime = zeros(NumVeh,1);
FillMsg = zeros(FillLen,1);
FillSeqNum = zeros(NumVeh,1);
FillMsgNum = zeros(NumVeh,1);

% TotMsgNum = StatMsgNum+ContDMsgNum;
TotMsgNum = zeros(NumVeh,1);

PartMsgNum = zeros(NumVeh,1);
PartMsgLen = zeros(NumVeh,1);
for v=1:NumVeh
    if(V1MsgCtIn(4,v) > 0)  % look at once memory instituted
        PartMsgNum(v,1) = 1;
        PartMsgLen(v,1) = V1MsgCtIn(4,v);
    end
    % TotMsgNum(v,1) = sum(MsgAttOut(:,3,v));
end

PartMsg = zeros(MaxLen,1,NumVeh);
PartMsgNew = zeros(MaxLen,1,NumVeh);  % look at once memory instituted
V1MsgRemOut = zeros(MaxLen,NumVeh);  % look at later
PartMsg(1:MaxLen,1,:) = V1MsgRemIn(1:MaxLen,:); % look at later

V1MsgCtOut = zeros(7,NumVeh);  % look at later

test = zeros(NumVeh, NumChan);  % generates VehTXst
for v=1:NumVeh
    for c = 1:NumChan
        if ChanVehMatBit(v,c) > 0 && v == 1
            VehTXst(1,c,v) = 1;
            test(v,c) = 1;
        elseif ChanVehMatBit(v,c) > 0 && v > 1
            VehTXst(1,c,v) = sum(ChanVehMatBit(1:(v-1),c))+1;
            test(v,c) = sum(ChanVehMatBit(1:(v-1),c))+1;
        end
    end
    TotMsgNum(v,1) = ceil(max(ChanVehMatBit(v,:))/FillLen);
end

ChanVehMatRem = ChanVehMatBit;  % remainder of unallocated
ChanVehMatBit is untouched

for v=1:NumVeh
    for c = 1:NumChan
        if ChanVehMatRem(v,c) > 0
            ChanVehMatRem(v,c) = ChanVehMatRem(v,c) -48;
        end
    end
```

```
end

% writes messages to allocated spaces
idx2 = 0;
idx = 0;
for v = 1:NumVeh
    for x = 1:NumChan % checking for all channels x=channel and column
        idx = 0;
        for z = 1:(TotMsgNum(v,1)+NumChan) %if messages to write or
last one pads // first space is for header
            idx2 = VehTXst(1,x,v)+47; % start and includes header
allocation
            r = ChanVehMatRem(v,x);
            if (r > 0 ) % if room on current channel

                if PartMsgNum(v,1) >= 1
                    MsgLen = PartMsgLen(v,1);  % MsgLen becomes old and
PartMsgLen becomes new PartMsgLen

                    if(MsgLen > r && r <= MaxLen) % if only partial
room for partial, break up message (MaxLen portion defines matrix size)

                        PartMsgNum(v,1) = 1; % checked
                        PartMsgLen(v,1) = MsgLen - r; % checked
                        PartMsgNew(1:PartMsgLen(v,1),1,v) =
PartMsg(r+1:MsgLen,1,v); %PartMsgNew takes place of PartMsg; PartMsg
takes place of StatMsg
                        VehTX(idx+idx2+1:idx+idx2+r,x,v) =
PartMsg(1:r,1,v);  %
                        PartMsg(:,1,v) = zeros(MaxLen,1);
                        PartMsg(1:PartMsgLen(v,1),1,v) =
PartMsgNew(1:PartMsgLen(v,1),1,v);
                        PartMsgNew(:,1,v) = zeros(MaxLen,1);
                        ChanVehMatRem(v,x) = 0;
                        idx = idx + r;

                    elseif MsgLen <= MaxLen %(MaxLen portion defines
matrix size)

                        VehTX(idx+idx2+1:idx+idx2+MsgLen,x,v)=
PartMsg(1:MsgLen,1,v);
                        PartMsgNum(v,1) = 0;
                        PartMsgLen(v,1) = 0;
                        PartMsg(:,1,v) = zeros(MaxLen,1);
                        ChanVehMatRem(v,x) = ChanVehMatRem(v,x) -
MsgLen;
                        idx = idx+MsgLen;

                    end

                elseif sum(MsgAttOut(:,3,v)) >=1%ContDMsgNum >=1

                    ContDSeqNum(v,1) = ContDSeqNum(v,1) +1;
                    %%%%%%%% below change important
```

```
                    ContDSeqNumTx = mod(ContDSeqNum(v,1),256); %
transmits 0-255; starts at 1; vice 1-255
                    SeqIdx = mod(ContDSeqNum(v,1)-1,BuffSize)+1;  %
verify +1; starts at 1, range 1-1024
                    MsgLen = MsgAttOut(SeqIdx,1,v);
                    MsgVehBi = de2bi(MsgAttOut(SeqIdx,4,v),4,'left-
msb')';
                    MsgTypeBi = de2bi(1,4,'left-msb')';
                    MsgLenBi = de2bi(MsgAttOut(SeqIdx,1,v)/8,16,'left-
msb')'; % and this
                    ContDSeqNumBi = de2bi(ContDSeqNumTx,8,'left-msb')';
% changed this too
                    ContDTimeBi = de2bi(MsgAttOut(SeqIdx,2,v),16,'left-
msb')'; % and this
                    MsgAttOut(SeqIdx,:,v) = [0 0 0 0];  % added this
                    ContDMsgHdr =
vertcat(MsgVehBi,MsgTypeBi,MsgLenBi,ContDSeqNumBi,ContDTimeBi);
                    MsgTemp(idx+idx2+1:idx+idx2+48,x) = ContDMsgHdr;


                    if(MsgLen > r && r <= MaxLen) % if only partial
room, break up message
                        PartMsgNum(v,1) = 1;
                        PartMsgLen(v,1) = MsgLen - r;
                        PartMsg(1:PartMsgLen(v,1),1,v) =
MsgTemp(idx+idx2+r+1:idx+idx2+MsgLen,x);  % new code y
                        VehTX(idx+idx2+1:(idx+idx2+r),x,v) =
MsgTemp(idx+idx2+1:idx+idx2+r,x); % new code y
                        ChanVehMatRem(v,x) = 0;
                        idx = idx + r;

                    elseif MsgLen <= MaxLen

                        VehTX(idx+idx2+1:idx+idx2+MsgLen,x,v)=
MsgTemp(idx+idx2+1:idx+idx2+MsgLen,x); %new code y
                        idx = idx+MsgLen;
                        ChanVehMatRem(v,x) = ChanVehMatRem(v,x) -
MsgLen;

                    else

                    end

                else % if left over space and nothing to say then pad
with zeros
                    % create pad message

                    MsgLen = 208;  % change to ChanVehMatRem(CurrVeh,x)
with minimum of FillMsgHdr Length when expand msglen and also ensure
divisible by 8
                    MsgVehBi = de2bi(v,4,'left-msb')';
                    MsgTypeBi = de2bi(9,4,'left-msb')';
                    MsgLenBi = de2bi(MsgLen/8,16,'left-msb')';
```

```
                    FillSeqNumBi = de2bi(FillSeqNum(v,1),8,'left-
msb')';
                    FillTimeBi = de2bi(CurrTime,16,'left-msb')';
                    FillMsgHdr =
vertcat(MsgVehBi,MsgTypeBi,MsgLenBi,FillSeqNumBi,FillTimeBi);
                    MsgTemp(idx+idx2+1:idx+idx2+48,x) = FillMsgHdr;

                    if(MsgLen > r) % if only partial room, break up
message
                        PartMsgNum(v,1) = 1;
                        PartMsgLen(v,1) = MsgLen - r;
                        PartMsg(1:PartMsgLen(v,1),1,v) =
MsgTemp(idx+idx2+r+1:idx+idx2+MsgLen,x); % new code y
                        VehTX(idx+idx2+1:idx+idx2+r,x,v) =
MsgTemp(idx+idx2+1:idx+idx2+r,x); % new code y
                        ChanVehMatRem(v,x) = 0;
                        idx = idx + r;

                    else


VehTX(idx+idx2+1:idx+idx2+MsgLen,x,v)=MsgTemp(idx+idx2+1:idx+idx2+MsgLe
n,x); % new code y
                        idx = idx+MsgLen;

                        ChanVehMatRem(v,x) = ChanVehMatRem(v,x) -
MsgLen;
                    end
                end
            end
        end
    end
end

HT = zeros(1,1);
EC = zeros(1,1);
Type = zeros(6,1);
LEN = zeros(16,1);
CID = zeros(16,1);
HCS = zeros(8,1);

HT = de2bi(0,1,'left-msb')';
EC = de2bi(1,1,'left-msb')';
Type = de2bi(4,6,'left-msb')'; % change for discard %%%%

HCS = de2bi(15,8,'left-msb')';  % not yet implememted

% append header if installed
for c=1:NumChan
    i=0;
    for v=1:NumVeh
        if(ChanVehMatBit(v,c)~=0)
            LEN = de2bi(((ChanVehMatBit(v,c))/8),16,'left-msb')';%
length in bytes including header, 2048 max, header is 6  %%%%
```

```
            CID = de2bi(v,16,'left-msb')';  %%%%%
            VehTX(i+1:i+48,c,v)=vertcat(HT,EC,Type,LEN,CID,HCS);
            i = i + ChanVehMatBit(v,c);
        end
    end
end
% create feedback loop for partial message from last transmission
% also add msg count to messages with feedback and reset at 2^8

for v=1:NumVeh
    V1MsgCtOut(4,v) = PartMsgLen(v,1);
    V1MsgCtOut(6,v) = ContDSeqNum(v,1);
    V1MsgCtOut(7,1) = CurrTime;
    V1MsgRemOut(1:PartMsgLen(v,1),v) = PartMsg(1:PartMsgLen(v,1),1,v);

end
```

5. BS MAC Transmitter

```
function [ VehCh1, VehCh2, VehCh3, VehCh4, VehCh5, VehCh6, VehCh7,
VehCh8,  VehCh1Rate, VehCh2Rate, VehCh3Rate, VehCh4Rate, VehCh5Rate,
VehCh6Rate, VehCh7Rate, VehCh8Rate,  VehCh1Snr, VehCh2Snr, VehCh3Snr,
VehCh4Snr, VehCh5Snr, VehCh6Snr, VehCh7Snr, VehCh8Snr, TXmatM, TXmatD]
= BSmacTX(ChanRateIDD, ChanRateIDU, ChanVehMatSnr, DLmsgIN, ULmsgIN,
VehTX)  % rate of transmission and transmission frame size
eml.extrinsic('de2bi');
% assume rateID to be static among chanel and for vehicle

% variable remaining static or input from higher level

BSnum = 1;
NumChan = 4;
NumVeh = 6;

ULalloLen = 28512;
ULsymb = 6336;
DLalloLen = 3456;
DLsymb = 768;
MAPlen = 384;
MAPsymb = 768;

HT = zeros(1,1);
EC = zeros(1,1);
Type = zeros(6,1);
RSV1 = zeros(1,1);
CI = zeros(1,1);
EKS = zeros(2,1);
RSV2= zeros(1,1);
LEN = zeros(11,1);
CID = zeros(16,1);
HCS = zeros(8,1);

MAPmat = zeros(MAPlen, NumChan);
```

```
TXmatD = zeros(DLalloLen,NumChan);
TXmatM = zeros(MAPlen,NumChan);

DLheaderMat2 = zeros(48,1);

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%TXmat should be written on by VehTX
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

i = 0;  % index for UL and DL MAP
i2 = 0; % index for DL headers
for c = 1:NumChan  % channel
    i = 0;
    i2 = 0;
    % there has to be a DLmap and UL map prefix even if CID not present
    MAPmat(1:8,c) = de2bi(c,8,'left-msb')'; % identifies uplink channel
%%%%%%%%%%
    MAPmat(9:16,c) = de2bi(ChanRateIDD(1,c),8,'left-msb')';  % burst
profile for channel
    MAPmat(17:64,c) = de2bi(BSnum,48,'left-msb')'; % BS ID
    for v1 = 1:NumVeh
        if (DLmsgIN (v1,c) ~= 0)  % if channel allocation matrix not
empty for spot create MAP
            if(v1 == 1)
                MAPmat(65+i:80+i,c) = de2bi(v1,16,'left-msb')'; %  CID
                MAPmat(81+i:96+i,c) = de2bi(MAPsymb+1,16,'left-msb')';
% Length
                i = i+32;
            else
                MAPmat(65+i:80+i,c) = de2bi(v1,16,'left-msb')'; %  CID
                MAPmat(81+i:96+i,c) =
de2bi(MAPsymb+1+sum(DLmsgIN(1:(v1-1),c)),16,'left-msb')';  % Length
                i = i+32;
            end
        end
    end

    if (i < 32)  % if nothing to write to DL MAP, leave first DL MAP
empty and move index up
        i = 32;
    end
    i = i+64;

    MAPmat(1+i:8+i,c) = de2bi(c,8,'left-msb')'; % identifies uplink
channel     %%%%%%%%%%%%
    MAPmat(9+i:16+i,c) = de2bi(ChanRateIDU(1,c),8,'left-msb')';  %
burst profile for channel
    MAPmat(17+i:64+i,c) = de2bi(BSnum,48,'left-msb')'; % BS ID
    MAPmat(65+i:96+i,c) = de2bi((DLsymb+MAPsymb+1),32,'left-msb')';
%Astartime
    for v2 = 1:NumVeh
        if (ULmsgIN (v2,c) ~= 0) % if channel allocation matrix not
empty for spot create MAP
            if (v2 == 1)
```

```
                MAPmat(97+i:112+i,c) = de2bi((v2+65280),16,'left-
msb')'; %  CID
                MAPmat(113+i:128+i,c) = de2bi(1,16,'left-msb')';  %
Length
                i=i+32;
            else
                MAPmat(97+i:112+i,c) = de2bi((v2+65280),16,'left-
msb')'; %  CID
                MAPmat(113+i:128+i,c) = de2bi(1+sum(ULmsgIN(1:(v2-
1),c)),16,'left-msb')';  % Length
                i=i+32;
            end
        end
    end

    TXmatM(1:MAPlen,c) = MAPmat(1:MAPlen,c);  % write MAP to TXmission

end

%%%%%%%%%%%%%%%%%% write messages inside DL allocation

TXmatD(1:DLalloLen,1) =
VehTX(1:DLalloLen,1,1)|VehTX(1:DLalloLen,1,2)|VehTX(1:DLalloLen,1,3)|Ve
hTX(1:DLalloLen,1,4)|VehTX(1:DLalloLen,1,5)|VehTX(1:DLalloLen,1,6);

TXmatD(1:DLalloLen,2) =
VehTX(1:DLalloLen,2,1)|VehTX(1:DLalloLen,2,2)|VehTX(1:DLalloLen,2,3)|Ve
hTX(1:DLalloLen,2,4)|VehTX(1:DLalloLen,2,5)|VehTX(1:DLalloLen,2,6);

TXmatD(1:DLalloLen,3) =
VehTX(1:DLalloLen,3,1)|VehTX(1:DLalloLen,3,2)|VehTX(1:DLalloLen,3,3)|Ve
hTX(1:DLalloLen,3,4)|VehTX(1:DLalloLen,3,5)|VehTX(1:DLalloLen,3,6);

TXmatD(1:DLalloLen,4) =
VehTX(1:DLalloLen,4,1)|VehTX(1:DLalloLen,4,2)|VehTX(1:DLalloLen,4,3)|Ve
hTX(1:DLalloLen,4,4)|VehTX(1:DLalloLen,4,5)|VehTX(1:DLalloLen,4,6);

VehCh = zeros(DLalloLen,2*NumChan);
VehChRate = zeros(1,2*NumChan);
VehChSnr = zeros(1,2*NumChan);

x =0;
for c = 1:NumChan
    for v = 1:NumVeh
        if (DLmsgIN(v,c) > 0)
            x = x+1;
            VehCh(:,x) = TXmatD(:,c)|TXmatD(:,c);
            VehChSnr(1,x) = ChanVehMatSnr(v,c);
            VehChRate(1,x) = ChanRateIDD(1,c);

        end
    end
end
```

```
VehCh1 = VehCh(:,1)|VehCh(:,1);
VehCh2 = VehCh(:,2)|VehCh(:,2);
VehCh3 = VehCh(:,3)|VehCh(:,3);
VehCh4 = VehCh(:,4)|VehCh(:,4);
VehCh5 = VehCh(:,5)|VehCh(:,5);
VehCh6 = VehCh(:,6)|VehCh(:,6);
VehCh7 = VehCh(:,7)|VehCh(:,7);
VehCh8 = VehCh(:,8)|VehCh(:,8);


VehCh1Rate = VehChRate(1,1);
VehCh2Rate = VehChRate(1,2);
VehCh3Rate = VehChRate(1,3);
VehCh4Rate = VehChRate(1,4);
VehCh5Rate = VehChRate(1,5);
VehCh6Rate = VehChRate(1,6);
VehCh7Rate = VehChRate(1,7);
VehCh8Rate = VehChRate(1,8);


VehCh1Snr = VehChSnr(1,1);
VehCh2Snr = VehChSnr(1,2);
VehCh3Snr = VehChSnr(1,3);
VehCh4Snr = VehChSnr(1,4);
VehCh5Snr = VehChSnr(1,5);
VehCh6Snr = VehChSnr(1,6);
VehCh7Snr = VehChSnr(1,7);
VehCh8Snr = VehChSnr(1,8);
```

C. SS SYSTEM

1. SS MAC Receiver

```
function [VehDataOut, ChanVehMatBitUL, MsgChanVehMat, MsgChanVehMatT2,
MsgChanVehLen]    = SSmacRX(VehDataIn, VehCh1, VehCh2, VehCh3, VehCh4,
VehCh5, VehCh6, VehCh7, VehCh8, RXmsgMatM, RXmsgMatD)  % rate of
transmission and transmission frame size

eml.extrinsic('bi2de', 'xor');

NumVeh = 6;
NumChan = 4;
ULalloLen = 28512;
ULsymb = 6336;
DLalloLen = 3456;
DLsymb = 768;
MAPlen = 384;
MAPsymb = 768;

RXmsgMatDcv = zeros(DLalloLen,NumChan,NumVeh);
VehCh = zeros(DLalloLen, 2*NumChan);
```

```
VehCh(:,1) = VehCh1;
VehCh(:,2) = VehCh2;
VehCh(:,3) = VehCh3;
VehCh(:,4) = VehCh4;
VehCh(:,5) = VehCh5;
VehCh(:,6) = VehCh6;
VehCh(:,7) = VehCh7;
VehCh(:,8) = VehCh8;

RXmsgMatDz = zeros(DLalloLen,1);

VehDataOut= VehDataIn;

%DL portion for all MAPs
%%%%%%%%

%Mapout1 becomes MAPmat(40,channel) with each column for a different
channel DL has first 20, UL has second
%inputFr becomes RXmsgMat(1:2048, channel)
MAPmat = zeros(40,4);

DMM = 0;
DCD = zeros(1,NumChan);
BSID = 0;
DCIDtest = 0;
DCIDmat = zeros(NumVeh*2+2,1); % temporarily holds the CID and start
positions

UMM = 0;
UCD = zeros(1,NumChan);
BSID = 0;
AstartTime = 0;
UCIDtest = 0;
UCIDmat= zeros(NumVeh*2+2,1);  % temporarily holds the CID and start
positions

for c=1:NumChan  % one large for loop looking at every channel
consecutively

    idx = 0;
    DMM = bi2de(RXmsgMatM(1:8,c)','left-msb'); % identifies uplink
channel
    DCD(1,c) = bi2de(RXmsgMatM(9:16,c)','left-msb');  % burst profile
for channel
    BSID = bi2de(RXmsgMatM(17:64,c)','left-msb'); % BS ID
    DCIDmat(1,1) = bi2de(RXmsgMatM(65:80,c)','left-msb'); % CONNECTION
ID

    if ( DCIDmat(1,1) ~= 0 ) % if chanel used; unused if zeros written
to first block
```

```
        DCIDmat(2,1) = bi2de(RXmsgMatM(81:96,c)','left-msb'); % start
time in ofdm symbols (individual)
        goflag = 1;
        DLnum = 1;

        for x=2:NumVeh

            DCIDtest =  bi2de(RXmsgMatM(97+idx:112+idx,c)','left-msb');
% if the UL portion of map end loop
            if(DCIDtest >= 256 )
                goflag = 0;
            end
            if(goflag == 1) % else record in matrix
                DCIDmat((x*2)-1,1) =
bi2de(RXmsgMatM(97+idx:112+idx,c)','left-msb');
                DCIDmat((x*2),1) =
bi2de(RXmsgMatM(113+idx:128+idx,c)','left-msb');
                DLnum = DLnum+1;
                idx = idx + 32;

            end
        end


MAPmat(1:4+2*DLnum,c)=vertcat(DMM,DCD(1,c),BSID,DLnum,DCIDmat(1:(DLnum*
2),1));  % record in matrix

    else

        DLnum = 0;
        MAPmat(1:4,c)=vertcat(DMM,DCD(1,c),BSID,DLnum);

    end

    idx = idx + 97;  % making it even # + idx: odd + idx ; resuming at
stopping point in MAP

    UMM = bi2de(RXmsgMatM(idx:idx+7,c)','left-msb'); % identifies
uplink channel
    UCD(1,c) = bi2de(RXmsgMatM(idx+8:idx+15,c)','left-msb');  % burst
profile for channel
    BSID = bi2de(RXmsgMatM(idx+16:idx+63,c)','left-msb'); % BS ID
    AstartTime = bi2de(RXmsgMatM(idx+64:idx+95,c)','left-msb');
    UCIDmat(1,1) = bi2de(RXmsgMatM(idx+96:idx+111,c)','left-msb'); %
CONNECTION ID


    if ( UCIDmat(1,1) ~= 0 ) % zeros written to spot if empty otherwise
continues
        UCIDmat(1,1) = UCIDmat(1,1)-65280;
        UCIDmat(2,1) = bi2de(RXmsgMatM(idx+112:idx+127,c)','left-msb');
% start time in ofdm symbols
```

```
        idx2 = idx+128;
        goflag=1;
        ULnum = 1;

        for x=2:NumVeh
            UCIDtest =  bi2de(RXmsgMatM(idx2:idx2+15,c)','left-msb');
            if(UCIDtest == 0 || idx2 >= MAPlen)  % if reaches end of
MAP (zeros) or index beyond range of map
                goflag = 0;
            end
            if(goflag == 1)
                UCIDmat((x*2)-1,1) =
bi2de(RXmsgMatM(idx2:idx2+15,c)','left-msb');   % record CID
                UCIDmat((x*2)-1,1) = UCIDmat((x*2)-1,1)-65280;% adjust
for 11111111 in front of CH
                UCIDmat((x*2),1) =
bi2de(RXmsgMatM(idx2+16:idx2+31,c)','left-msb');  % record start
                ULnum = ULnum+1;  % document number of UL allocations
                idx2 = idx2 + 32;
            end
        end


MAPmat(21:25+2*ULnum,c)=vertcat(UMM,UCD(1,c),BSID,AstartTime,ULnum,UCID
mat(1:(ULnum*2),1));

    else  % if zero UL allocations

        ULnum = 0;
        MAPmat(21:25,c)=vertcat(UMM,UCD(1,c),BSID,AstartTime,ULnum);

    end
end

% once DL messages are encoded, they can be decoded using below code

%%%% for decoding DL messages

%%%%%%%%%%%%% steps, create CHVeh

VehTXstUL = zeros(1,NumChan, NumVeh);  % start place includes header
(yet to be added)
VehTXstDL = zeros(1,NumChan, NumVeh);  % start place includes header
(yet to be added)

MAPmatDL = zeros(NumChan*2,20);
MAPmatUL = zeros(NumChan*2,20);

%separates MAPmat into UP and DL portions

for c = 1:NumChan
    MAPmatDL(c,1:20) = MAPmat(1:20,c);
    MAPmatUL(c,1:20) = MAPmat(21:40,c);
```

```
end

% reconstitute MsgIn / ChanVehMat for UL

ChanVehMatUL = zeros(NumVeh,NumChan);
ChanVehMatDL = zeros(NumVeh,NumChan);

numUL = 0;
numDL = 0;

SymBitMat = [.5 1 1.5 2 3 4 4.5];

ChanRateIDU = UCD + 1;

ChanRateIDD = DCD + 1;

% must reconvert symbols into bits based on rateID and subtracting 16
symbols (in addition to the 48 bits)

for x=1:NumChan
    if (MAPmatUL(x,5) == 0) % if empty do nothing
        z=z+1;
    else
        numUL = MAPmatUL(x,5); % if not empty then

        for y=1:NumVeh % maximum for Num of Veh
            if( y < numUL ) %extract down row of MAP
                % correct to turn into bits and for header

ChanVehMatUL(MAPmatUL(x,(4+y*2)),x)=MAPmatUL(x,(5+(y+1)*2))-
MAPmatUL(x,(5+y*2));
                VehTXstUL(1,x,MAPmatUL(x,(4+y*2))) =
MAPmatUL(x,(5+y*2));

            elseif (y == numUL) % for last one, different rules
                ChanVehMatUL(MAPmatUL(x,(4+y*2)),x)=ULsymb+1-
MAPmatUL(x,(5+y*2)); % took out 1025 and replaced with ULsymb+1
                VehTXstUL(1,x,MAPmatUL(x,(4+y*2))) =
MAPmatUL(x,(5+y*2));

            end
        end
    end
end

ChanVehMatBitUL = zeros(NumVeh, NumChan);
for c = 1:NumChan
    ChanVehMatBitUL(:,c) =
ChanVehMatUL(:,c)*SymBitMat(1,ChanRateIDU(1,c));
end

% need to account for header allocation
```

```
for c = 1:4  % deletes last 8 bits for convolution from each channel
and 48 bit header from
    for v = 1:NumVeh
        if ChanVehMatBitUL(v,c) > 0 && ChanVehMatBitUL(v,c) >= 48
            ChanVehMatBitUL(v,c) = ChanVehMatBitUL(v,c) - 48;
        elseif ChanVehMatBitUL(v,c) > 0 && ChanVehMatBitUL(v,c) < 48
            z=5;
        end
        if ChanVehMatBitUL(v,c) > 0 && v == NumVeh
            ChanVehMatBitUL(v,c) = ChanVehMatBitUL(v,c) - 8;
        elseif ChanVehMatBitUL(v,c) > 0 &&
sum(ChanVehMatBitUL(v+1:NumVeh,c)) == 0
            ChanVehMatBitUL(v,c) = ChanVehMatBitUL(v,c) - 8;
        end
    end
end

for x=1:NumChan
    if (MAPmatDL(x,4) == 0) % if empty do nothing
        z=z+1;
    else
        numDL = MAPmatDL(x,4); % if not empty then

        for y=1:NumVeh % maximum for Num of Veh
            if( y < numDL ) %extract down row of MAP
                % correct to turn into bits and for header

ChanVehMatDL(MAPmatDL(x,(3+y*2)),x)=MAPmatDL(x,(4+(y+1)*2))-
MAPmatDL(x,(4+y*2));
                VehTXstDL(1,x,MAPmatDL(x,(3+y*2))) =
MAPmatDL(x,(4+y*2));

            elseif (y == numDL) % for last one, different rules
                ChanVehMatDL(MAPmatDL(x,(3+y*2)),x)=DLsymb+MAPsymb+1-
MAPmatDL(x,(4+y*2)); % took out 1025 and replaced with DLsymb+1
                VehTXstDL(1,x,MAPmatDL(x,(3+y*2))) =
MAPmatDL(x,(4+y*2));
            end
        end
    end
end

ChanVehMatBitDL = zeros(NumVeh, NumChan);

for c = 1:NumChan
    ChanVehMatBitDL(:,c) =
ChanVehMatDL(:,c)*SymBitMat(1,ChanRateIDD(1,c));
end

for c = 1:4  % deletes last 8 bits for convolution from each channel no
need for header accounting on DL
    for v = 1:NumVeh
        if ChanVehMatBitDL(v,c) > 0 && v == NumVeh
            ChanVehMatBitDL(v,c) = ChanVehMatBitDL(v,c) - 8;
```

```
        elseif ChanVehMatBitDL(v,c) > 0 &&
sum(ChanVehMatBitDL(v+1:NumVeh,c)) == 0
            ChanVehMatBitDL(v,c) = ChanVehMatBitDL(v,c) - 8;
        end
    end
end

% Create Raw Matix of DL Messages

FrLen = 0;
VehID = 0;

MsgChanVehMat = zeros(DLalloLen,4,NumVeh);
MsgChanVehMatT1 = zeros(DLalloLen,4,NumVeh);
MsgChanVehMatT2 = zeros(DLalloLen,4,NumVeh);
MsgChanVehLenT = zeros(1,4,NumVeh);
MsgChanVehLen = zeros(1,4,NumVeh);

g = 0;

FrCount = zeros(NumVeh,1);
FrLoss = zeros(NumVeh,1);
throughput = zeros(NumVeh,1);
biterror = zeros(NumVeh,1);

x =0;
for c = 1:NumChan
    for v = 1:NumVeh
        if (ChanVehMatDL(v,c) > 0)
            x = x+1;
            RXmsgMatDcv(:,c,v) = VehCh(:,x);

        end
    end
end

idx = 0;
for c=1:NumChan
    idx = 0;
    for v=1:NumVeh
        if ChanVehMatBitDL(v,c) > 0  % using SS derived for now until
have the matchup right (has 48 less than BS info)
            g =
sum(xor(RXmsgMatD(idx+1:idx+48,c),RXmsgMatDcv(idx+1:idx+48,c,v)));  %
error checking
            FrCount(v,1) = FrCount(v,1) + 1;
            FrLen = bi2de(RXmsgMatD(idx+9:idx+24,c)','left-msb');  %
read length
            FrLen = FrLen*8;  % change to bits
            VehID = bi2de(RXmsgMatD(idx+25:idx+40,c)','left-msb');
% read cid
            if (FrLen <= DLalloLen)  % upper limit only for MATLAB
considerations
```

```
                if ((FrLen == (ChanVehMatBitDL(v,c))) && VehID == v)
% corroborate cid; corroborate lenght
                    MsgChanVehMat(1:(FrLen-48),c,v) =
RXmsgMatD(idx+49:idx+FrLen,c);              % write to Matrix without
header
                    MsgChanVehMatT1(1:(FrLen-48),c,v) =
RXmsgMatDcv(idx+49:idx+FrLen,c,v);       %%%%%%%%%%       % for
throughput and error detection
                    MsgChanVehLenT(1,c,v) = FrLen;  %%%%%%%%%%       %
for throughput
                    MsgChanVehLen(1,c,v) = FrLen - 48;  % shorten to
without header allotment
                end
            end
            if g >= 1 && (FrLen <= DLalloLen)  % upper limit only for
MATLAB considerations
                % counts number of errors in header and zeros out frame
if too many errors for packet loss

                FrLoss(v,1) = FrLoss(v,1) + 1;
                MsgChanVehMatT2(1:(FrLen-48),c,v) =
RXmsgMatDz(idx+49:idx+FrLen,1);
            elseif (FrLen <= DLalloLen)
                MsgChanVehMatT2(1:(FrLen-48),c,v) =
RXmsgMatDcv(idx+49:idx+FrLen,c,v);  % otherwise just writes transmitted
info
            end
            idx = idx + (ChanVehMatBitDL(v,c));
        end
    end
end

FrCountData = 5;
FrLossData = 6;
throughputData = 2;
biterrorData = 3;

for v=1:NumVeh

    biterror(v,1) =
sum(sum(xor(MsgChanVehMatT1(:,:,v),MsgChanVehMat(:,:,v))));
    throughput(v,1) = sum(MsgChanVehLenT(1,:,v));

    VehDataOut(v,FrCountData) = FrCount(v,1);
    VehDataOut(v,throughputData) =throughput(v,1);
    VehDataOut(v,biterrorData) =biterror(v,1);
    VehDataOut(v,FrLossData) = FrLoss(v,1);

end
```

2. SS MAC Assembly

```
function [VehDataOut, MsgRemMatOut, MsgRemMatOutT, MsgRemLenOut,
MsgSeqOut, RelMsgMatDL, RelMsgNumDL]  = SSmacASSY(VehDataIn,
MsgChanVehMat, MsgChanVehMatT, MsgChanVehLen, Time, MsgRemMatIn,
MsgRemMatInT, MsgRemLenIn, MsgSeqIn)  % rate of transmission and
transmission frame size

eml.extrinsic('bi2de');

eml.extrinsic('de2bi');

NumVeh = 6;
NumChan = 4;

ContDLen = 560;
FillLen = 208;
ULalloLen = 28512;
ULsymb = 6336;
DLalloLen = 3456;
DLsymb = 768;
MAPlen = 384;
MAPsymb = 768;

RelMsgMatDL = zeros(5,NumVeh,64,2);
RelMsgNumDL = zeros(5,NumVeh);

MsgRemMatOut = zeros(ContDLen,NumVeh);
MsgRemMatOutT = zeros(ContDLen,NumVeh);
MsgRemLenOut = zeros(1,NumVeh);

MsgRemMat = zeros(ContDLen,NumVeh);
MsgRemMatT = zeros(ContDLen,NumVeh);
MsgRemLen = zeros(1,NumVeh);

MsgRemMat = MsgRemMatIn;
MsgRemMatT = MsgRemMatInT;
MsgRemLen = MsgRemLenIn;

ContDCount = zeros(NumVeh,1);
MsgBuff = zeros(ContDLen+NumChan*DLalloLen,1);  % full possible length
+ remainder poss
MsgBuffT = zeros(ContDLen+NumChan*DLalloLen,1);
MsgBuffLen = length(MsgBuff);

MsgSeqOut = MsgSeqIn;
MsgVeh = 0;
MsgType = 0;
MsgLen = 0;
MsgSeq = 0;
MsgTime = 0;

i = 0; % last bit of message
```

```
i2 = 0; % progress in reading message
z = 0;
testz = 0;
g = 0;
packetloss = zeros(NumVeh,1);
packetcount = zeros(NumVeh,1);
goodput = zeros(NumVeh,1);
packetdelay = zeros(NumVeh,1);
maxpacketdelay = zeros(NumVeh,1);
n1=0;

for v=1:NumVeh

    n1=0;
    i = 0;
    i2 = 0;
    if MsgRemLen(1,v) > 0
        MsgBuff(1+i:MsgRemLen(1,v)+i,1) =
MsgRemMat(1:MsgRemLen(1,v),v);
        MsgBuffT(1+i:MsgRemLen(1,v)+i,1) =
MsgRemMatT(1:MsgRemLen(1,v),v);  % for packet loss checking
        i = i + MsgRemLen(1,v);
    end

    for c=1:NumChan
        z = MsgChanVehLen(1,c,v);
        if z > 0 && z <= DLalloLen
            MsgBuff(1+i:z+i,1) = MsgChanVehMat(1:z,c,v);  % create full
length msg
            MsgBuffT(1+i:z+i,1) = MsgChanVehMatT(1:z,c,v);  % for
packet loss checking
            i = i + z;
        end
        % i symbolizes the last bit (length), and the first bit should
be the start of a header

    end
    % i2 is progress of reading

    MaxNum = ceil(MsgBuffLen/FillLen);

    for x = 1:MaxNum

        if (i-i2 >= 48) % if at least beginning of header sent it is a
message
            MsgVeh = bi2de(MsgBuff(i2+1:i2+4,1)','left-msb');
            MsgType = bi2de(MsgBuff(i2+5:i2+8,1)','left-msb');
            MsgLen = bi2de(MsgBuff(i2+9:i2+24,1)','left-msb');
            MsgSeq = bi2de(MsgBuff(i2+25:i2+32,1)','left-msb');
            MsgTime = bi2de(MsgBuff(i2+33:i2+48,1)','left-msb');
            MsgLen = MsgLen*8;
            g = sum(xor(MsgBuffT(i2+1:i2+48,1),MsgBuff(i2+1:i2+48,1)));
% error checking
```

```
            if (i-i2 >= MsgLen && MsgLen >0 && MsgType > 0) % if
complete message, break down message and record new i2

                if(MsgType == 1)
                    ContDCount(v,1) =  ContDCount(v,1) + 1;
                    if (MsgSeq < 257 && (MsgSeq ==
(MsgSeqIn(v,1)+ContDCount(v,1)) || MsgSeq ==
(MsgSeqIn(v,1)+ContDCount(v,1))-256) )
                        MsgSeqOut(v,1) = MsgSeq;
                    elseif (MsgSeq == 1 &&
(MsgSeqIn(v,1)+ContDCount(v,1)) == 257)
                        MsgSeqOut(v,1) = MsgSeq;
                    else
                        MsgSeqOut(v,1) = MsgSeq;

                    end

                    if g >= 1 || n1 > 63 % error checking
                        packetloss(MsgVeh,1) = packetloss(MsgVeh,1) +
1;

                    else % send it to next hop
                        n1 = n1+1;
                        RelMsgMatDL(MsgType,v,n1,1) = MsgTime;
                        RelMsgMatDL(MsgType,v,n1,2) = MsgLen;
                        goodput(v,1) = goodput(v,1) + MsgLen - 48;

                    end

                    packetcount(MsgVeh,1) = packetcount(MsgVeh,1) + 1;
                    packetdelay(MsgVeh,MsgType) =
packetdelay(MsgVeh,MsgType) + mod(Time-MsgTime,65536); %%change
                    if mod(Time-MsgTime,65536) >
maxpacketdelay(MsgVeh,MsgType)
                        maxpacketdelay(MsgVeh,MsgType) = mod(Time-
MsgTime,65536);

                    end

                elseif(MsgType == 9)

                    testz = 4;

                else

                end
                i2 = i2 + MsgLen;  % i2 is progress in reading message

            else  % if incomplete message%write to MsgRem
                MsgRemMatOut(1:(i-i2),v) = MsgBuff(i2+1:i,1);
                MsgRemMatOutT(1:(i-i2),v) = MsgBuffT(i2+1:i,1);  %
error checking
                MsgRemLenOut(1,v) = i-i2;
```

```
                i2 = i; % new
            end

        elseif(i-i2 > 0)  % if incomplete message (only part of
header)%write to MsgRem
            MsgRemMatOut(1:(i-i2),v) = MsgBuff(i2+1:i,1);
            MsgRemMatOutT(1:(i-i2),v) = MsgBuffT(i2+1:i,1);  % error
checking
            MsgRemLenOut(1,v) = i-i2;
            i2 = i; % new

        end
    end
    RelMsgNumDL(1,v) = n1;
end

GoodputData = 4;
PkCountContDData = 8;
PkLossData = 12;
PkMaxDelayContDData = 24;
PkAvgDelayContDData = 20;

VehDataOut= VehDataIn;

for v = 1:NumVeh

    VehDataOut(v,GoodputData) = goodput(v,1);
    VehDataOut(v,PkCountContDData) =  packetcount(v,1);
    VehDataOut(v,PkLossData) =  packetloss(v,1);
    VehDataOut(v,PkMaxDelayContDData) =  maxpacketdelay(v,1);
    VehDataOut(v,PkAvgDelayContDData) = packetdelay(v,1);

end
```

3. SS MAC Controller

```
function [VehTX,packetdrop,V1MsgCtOut, V1MsgRemOut,MsgAttOut] =
SSmacCTRL(inFR1,inFR2, inFR3, inFR4, RadEn, VidEn, ChanVehMatUL,
V1MsgCtIn, V1MsgRemIn, MsgAttIn, RelMsgMat, RelMsgNum, UAVatt)  % rate
of transmission and transmission frame size
eml.extrinsic('de2bi');

NumChan = 4;
NumVeh = 6;
CurrVeh = 1;

VehNum = 1;
VehPri = 2;
VehContD = 3;
VehStatU = 4;
VehContU = 5;
```

```
VehRadU = 6;
VehVidU = 7;

MaxRelayNum = 3;

VehRelaying = zeros(MaxRelayNum,1);

RelayNum = 0;
for v=1:MaxRelayNum
    if UAVatt(CurrVeh,7+v) > 0
        RelayNum = RelayNum+1;
        VehRelaying(RelayNum,1) = UAVatt(CurrVeh,7+v);
    end
end

testz = 0;
ULalloLen = 28512;
ULsymb = 6336;
DLalloLen = 3456;
DLsymb = 768;
MAPlen = 384;
MAPsymb = 768;

BuffSize = 1024;  % changed from 1024 for only a 4 second buffer

MaxPartLen = 1000;
MaxLen = 524280;  % add MaxLen of 1000 with MsgTemp that long  52432;
MaxRef = ULalloLen+MaxPartLen;

MsgTemp = zeros(ULalloLen+MaxPartLen,NumChan);% 29148 for now
MsgTemp(:,1) = inFR1(1:MaxRef,1);
MsgTemp(:,2) = inFR2(1:MaxRef,1);
MsgTemp(:,3) = inFR3(1:MaxRef,1);
MsgTemp(:,4) = inFR4(1:MaxRef,1);

VehTX=zeros(ULalloLen,4);  % Vehicle TX according to channel
VehTXst = zeros(1,NumChan);  % start place includes header (yet to be
added)

ChanVehMat = zeros(NumVeh,NumChan);

ChanVehMat = ChanVehMatUL;

ChanVehMatSt = ChanVehMatUL;

for v = 1:NumVeh
    for c = 1:NumChan  % need to compensate for header
        if ChanVehMatSt(v,c) > 0

            ChanVehMatSt(v,c) = ChanVehMatSt(v,c) + 48;
        end
    end
end
```

```
test = zeros(NumVeh, NumChan);  % creates VehTXst

for c = 1:NumChan  % need to compensate for header
    if ChanVehMatSt(CurrVeh,c) > 0 && CurrVeh == 1
        VehTXst(1,c) = 1;
        test(CurrVeh,c) = 1;

    elseif ChanVehMatSt(CurrVeh,c) > 0 && CurrVeh > 1
        VehTXst(1,c) = sum(ChanVehMatSt(1:(CurrVeh-1),c))+1;
        test(CurrVeh,c) = sum(ChanVehMatSt(1:(CurrVeh-1),c))+1;
        ChanVehMatSt(CurrVeh,c) = ChanVehMatSt(CurrVeh,c) + 48;
    end
end

MsgAttOut = zeros(BuffSize,4,5);
MsgAttOut = MsgAttIn;
CurrTime = mod(V1MsgCtIn(10,1)+1,65536);
StatUMsgNum = 1;  % only transmit one StatMsg
ContUMsgNum = 0;
RadMsgNum = 0;
VidMsgNum = 0;

if (mod(CurrTime+CurrVeh,7) == 0)
    ContUMsgNum = 1;
end

if (mod(CurrTime+CurrVeh,500) == 0) && RadEn == 1
    RadMsgNum = 1;
end

if (mod(CurrTime*3+CurrVeh,40) < 3) && VidEn == 1
    VidMsgNum = 1;
end

MsgLen = 0;

MsgVehBi = zeros(4,1);
MsgTypeBi = zeros(4,1);
MsgLenBi = zeros(16,1);
StatUTypeBi = zeros(8,1);
StatUNumBi = zeros(8,1);
StatUSeqNumBi = zeros(8,1);
StatUTimeBi = zeros(16,1);
ContUTypeBi = zeros(8,1);
ContUNumBi = zeros(8,1);
ContUByteBi = zeros(16,1);
ContUSeqNumBi = zeros(8,1);
ContUTimeBi = zeros(16,1);
RadTypeBi = zeros(8,1);
RadNumBi = zeros(8,1);
RadByteBi = zeros(16,1);
RadSeqNumBi = zeros(8,1);
RadTimeBi = zeros(16,1);
```

```
VidTypeBi = zeros(8,1);
VidNumBi = zeros(8,1);
VidByteBi = zeros(16,1);
VidSeqNumBi = zeros(8,1);
VidTimeBi = zeros(16,1);
FillTypeBi = zeros(8,1);
FillNumBi = zeros(8,1);
FillByteBi = zeros(16,1);
FillSeqNumBi = zeros(8,1);
FillTimeBi = zeros(16,1);
% StatUMsgPadBi = zeros(40,1);
ContUMsgPadBi = zeros(80,1);
RadMsgPadBi = zeros(80,1);
VidMsgPadBi = zeros(160,1);
FillMsgPadBi = zeros(160,1);
PadFil = zeros(ULalloLen,1);
StatULen = 144;
StatUMsg = zeros(StatULen,1);
StatUNewMsg = StatUMsgNum;   % added this
StatUSeqNum = V1MsgCtIn(5,1);

%%%%%%%%%%%%%% Add Message to queue
StatUMsgCt = sum(MsgAttOut(:,3,2));
if StatUNewMsg + StatUMsgCt > BuffSize
    StatUNewMsg = BuffSize - StatUMsgCt;
end

if StatUNewMsg > 0
    for x=1:StatUNewMsg
        i = mod(StatUSeqNum-1+x,BuffSize)+1; % starts at 1; range 1-
1024  %+StatUMsgCt taken out
        %%% enter length
        MsgAttOut(i,1,2) = StatULen;
        %%% enter time
        MsgAttOut(i,2,2) = CurrTime;
        %%% enter count
        MsgAttOut(i,3,2) = 1;
        %%%% enter vehicle
        MsgAttOut(i,4,2) = CurrVeh;

    end
end

MessageTypes = 5;

packetdrop = zeros(1,MessageTypes);

ContULen = 128;                  %%%%%%%%%%%%%%%%%%%%%%%% arrange
V1MsgCt and V1MsgRem for all messages below

ContUMsgHdr = zeros(48,1);

ContUNewMsg = ContUMsgNum;   % added this
ContUSeqNum = V1MsgCtIn(6,1);  % last transmitted
```

```
%%%%%%%%%%%%%% Add Message to queue
ContUMsgCt = sum(MsgAttOut(:,3,3));
if ContUNewMsg + ContUMsgCt > BuffSize
    packetdrop(1,3) = ContUNewMsg + ContUMsgCt - BuffSize;
    ContUNewMsg = BuffSize - ContUMsgCt;
end

if ContUNewMsg > 0
    for x=1:ContUNewMsg
        i = mod(ContUSeqNum-1+ContUMsgCt+x,BuffSize)+1; % starts at 1;
range 1-1024
        %%% enter length
        MsgAttOut(i,1,3) = ContULen;
        %%% enter time
        MsgAttOut(i,2,3) = CurrTime;
        %%% enter count
        MsgAttOut(i,3,3) = 1;
        %%%% enter vehicle
        MsgAttOut(i,4,3) = CurrVeh;
    end
end

for r=1:MaxRelayNum
    if VehRelaying(r,1) > 0
        ContUNewMsgRel = RelMsgNum(3,VehRelaying(r,1));
        ContUMsgCtRel = sum(MsgAttOut(:,3,3));
        if ContUNewMsgRel + ContUMsgCtRel > BuffSize
            packetdrop(1,3) = ContUNewMsgRel + ContUMsgCtRel -
BuffSize;
            ContUNewMsgRel = BuffSize - ContUMsgCtRel;
        end

        if ContUNewMsgRel > 0
            for x=1:ContUNewMsgRel
                i = mod(ContUSeqNum-1+x+ContUMsgCtRel,BuffSize)+1; %
starts at 1; range 1-1024  %+StatUMsgCt taken out
                %%% enter length
                MsgAttOut(i,1,3) = ContULen;
                %%% enter time
                MsgAttOut(i,2,3) = RelMsgMat(3,VehRelaying(r,1),x,1);
                %%% enter count
                MsgAttOut(i,3,3) = 1;
                %%%% enter vehicle
                MsgAttOut(i,4,3) = VehRelaying(r,1);
            end
        end
    end
end

ContUMsgNumTx = sum(MsgAttOut(:,3,3));
if(ContUMsgNumTx >255)
    ContUMsgNumTx =255;
end
```

```
ContUByteNumTx = ceil(sum(MsgAttOut(:,1,3))/8);
if(ContUByteNumTx >65535)
    ContUByteNumTx =65535;
elseif ContUByteNumTx  == 0
    ContUByteNumTx = 8;
end

RadLen = [5920  5968    6024    6080    6136    6200    6256    6320
6376    6440    6504    6576    6640    6712    6784    6856    6928
7000    7080    7160    7240    7320    7408    7496    7584    7672
7768    7864    7960    8064    8168    8280    8384    8504    8616
8736    8864    8992    9120    9256    9400    9544    9688    9848
10008   10168   10344   10520   10704   10896   11096   11296   11512
11736   11968   12216   12464   12736   13008   13304   13608   13928
14264   14616   14984   15376   15792   16240   16704   17200   17728
18296   18896   19544   20248   21000   21816   22696   23656   24712
25872   27152   28576   30168   31960   33992   36320   39016   42168
45920   50448   56048   63152   72472   85280   104072  464400  524232
524232  524232];

RadMsgHdr = zeros(48,1);
RadNewMsg = RadMsgNum;   % added this
RadSeqNum =  V1MsgCtIn(7,1);

%%%%%%%%%%%%%% Add Message to queue
RadMsgCt = sum(MsgAttOut(:,3,4));
if RadNewMsg + RadMsgCt > BuffSize
    packetdrop(1,4) = RadNewMsg + RadMsgCt - BuffSize;
    RadNewMsg = BuffSize - RadMsgCt;
end

if RadNewMsg > 0
    for x=1:RadNewMsg
        i = mod(RadSeqNum-1+RadMsgCt+x,BuffSize)+1; % starts at 1;
range 1-1024
        %%% enter length
        MsgLen = RadLen(1,ceil(rand(1)*100));

        if MsgLen > MaxLen
            MsgLen = MaxLen;
        end

        MsgAttOut(i,1,4) = MsgLen;
        %%% enter time
        MsgAttOut(i,2,4) = CurrTime;
        %%% enter count
        MsgAttOut(i,3,4) = 1;
        %%%% enter vehicle
        MsgAttOut(i,4,4) = CurrVeh;

    end
end

for r=1:MaxRelayNum
```

```
    if VehRelaying(r,1) > 0

        RadNewMsgRel = RelMsgNum(4,VehRelaying(r,1));
        RadMsgCtRel = sum(MsgAttOut(:,3,4));
        if RadNewMsgRel + RadMsgCtRel > BuffSize
            packetdrop(1,4) = RadNewMsgRel + RadMsgCtRel - BuffSize;
            RadNewMsgRel = BuffSize - RadMsgCtRel;
        end

        if RadNewMsgRel > 0
            for x=1:RadNewMsgRel
                i = mod(RadSeqNum-1+x+RadMsgCtRel,BuffSize)+1; % starts
at 1; range 1-1024  %+StatUMsgCt taken out
                %%% enter length
                MsgAttOut(i,1,4) = RelMsgMat(4,VehRelaying(r,1),x,2);
                %%% enter time
                MsgAttOut(i,2,4) = RelMsgMat(4,VehRelaying(r,1),x,1);
                %%% enter count
                MsgAttOut(i,3,4) = 1;
                %%%% enter vehicle
                MsgAttOut(i,4,4) = VehRelaying(r,1);

            end
        end
    end
end


RadMsgNumTx = sum(MsgAttOut(:,3,4));
if(RadMsgNumTx >255)
    RadMsgNumTx =255;
end
RadByteNumTx = ceil(sum(MsgAttOut(:,1,4))/8);
if(RadByteNumTx >65535)
    RadByteNumTx =65535;
elseif RadByteNumTx  == 0
    RadByteNumTx = 8;

end

VidMsgHdr = zeros(48,1);
VidNewMsg = VidMsgNum;   % added this
VidSeqNum =  V1MsgCtIn(8,1);
VidLen = [8576  8648    8728    8816    8896    8976    9064    9152
9240    9336    9432    9528    9624    9720    9824    9928    10040
10144   10256   10376   10488   10608   10736   10856   10992   11120
11256   11400   11544   11688   11840   11992   12152   12320   12488
12664   12840   13024   13216   13416   13616   13824   14040   14264
14496   14736   14984   15248   15512   15792   16080   16376   16688
17008   17352   17704   18072   18456   18856   19272   19720   20184
20664   21176   21720   22288   22896   23528   24208   24928   25696
26512   27392   28336   29344   30432   31616   32896   34296   35824
37504   39360   41424   43728   46328   49272   52648   56552   61120
```

```
66560   73128   81240   91536   105048  524280  524280  524280  524280
524280  524280];
VidBuffSize = 64;  % limits to 1 second delay

clearbuffer = zeros(1024,4);

%%%%%%%%%%%%%% Add Message to queue
VidMsgCt = sum(MsgAttOut(:,3,5));
if VidNewMsg + VidMsgCt > VidBuffSize
    MsgAttOut(:,:,5) = clearbuffer;
    VidMsgCt = 0;
    packetdrop(1,5) = VidBuffSize;
end

if VidNewMsg > 0
    for x=1:VidNewMsg
        i = mod(VidSeqNum-1+VidMsgCt+x,BuffSize)+1; % starts at 1;
range 1-1024
        %%% enter length
        MsgLen = VidLen(1,ceil(rand(1)*100));

        if MsgLen > MaxLen
            MsgLen = MaxLen;
        end

        MsgAttOut(i,1,5) = MsgLen;  %320 + 8.*round(randn);
        %%% enter time
        MsgAttOut(i,2,5) = CurrTime;
        %%% enter count
        MsgAttOut(i,3,5) = 1;
        %%%% enter vehicle
        MsgAttOut(i,4,5) = CurrVeh;
    end
end

for r=1:MaxRelayNum

    if VehRelaying(r,1) > 0

        VidNewMsgRel = RelMsgNum(5,VehRelaying(r,1));
        VidMsgCtRel = sum(MsgAttOut(:,3,5));
        if VidNewMsgRel + VidMsgCtRel > BuffSize
            packetdrop(1,5) = VidNewMsgRel + VidMsgCtRel - BuffSize;
            VidNewMsgRel = BuffSize - VidMsgCtRel;
        end

        if VidNewMsgRel > 0
            for x=1:VidNewMsgRel
                i = mod(VidSeqNum-1+x+VidMsgCtRel,BuffSize)+1; % starts
at 1; range 1-1024  %+StatUMsgCt taken out
                %%% enter length
                MsgAttOut(i,1,5) = RelMsgMat(5,VehRelaying(r,1),x,2);
                %%% enter time
                MsgAttOut(i,2,5) = RelMsgMat(5,VehRelaying(r,1),x,1);
```

```
                %%% enter count
                MsgAttOut(i,3,5) = 1;
                %%%% enter vehicle
                MsgAttOut(i,4,5) = VehRelaying(r,1);

            end
        end
    end
end

VidMsgNumTx = sum(MsgAttOut(:,3,5));
if(VidMsgNumTx >255)
    VidMsgNumTx =255;
end
VidByteNumTx = ceil(sum(MsgAttOut(:,1,5))/8);
if(VidByteNumTx >65535)
    VidByteNumTx =65535;
elseif VidByteNumTx  == 0 && (RelayNum > 0 || VidEn > 0)
    VidByteNumTx = 400;
elseif VidByteNumTx  == 0
    VidByteNumTx = 8;
end

FillLen = 208;
FillMsg = zeros(FillLen,1);
FillMsgHdr = zeros(48,1);
FillMsgNum = 0;%abs(FillMsgNum);
FillSeqNum =  0;%V1MsgCtIn(8,1);
TotMsgNum = ceil(max(ChanVehMat(CurrVeh,:))/100);
PartMsgNum = 0;
PartMsgLen = 0;
if(V1MsgCtIn(9,1) > 0)
    PartMsgNum = 1;
    PartMsgLen = V1MsgCtIn(9,1);
end

PartMsg = zeros(MaxPartLen,1);
PartMsgNew = zeros(MaxPartLen,1);
V1MsgRemOut = zeros(MaxPartLen,1);
PartMsg(1:MaxPartLen,1) = V1MsgRemIn(1:MaxPartLen,1);

V1MsgCtOut = zeros(10,1);
ChanVehMatRem = ChanVehMat;  % remainder of unallocated
MsgCount = 0;  % eventually place message count in indiv. message
header

% writes messages to allocated spaces

idx = 0;
idx2 = 0;

for x = 1:NumChan % checking for all channels x=channel and column
    idx = 0;
```

```
    for z = 1:(TotMsgNum+NumChan) %if messages to write or last one
pads // first space is for header
        idx2 = VehTXst(1,x)+47; % start and includes header allocation
        r = ChanVehMatRem(CurrVeh,x);

        if (r > 0 ) % if room on current channel
            MsgCount = MsgCount+1;

            if PartMsgNum >= 1
                MsgLen = PartMsgLen;  % MsgLen becomes old and
PartMsgLen becomes new PartMsgLen

                if MsgLen > MaxPartLen

                    if(MsgLen > r && r <= MaxRef) % if only partial
room for partial, break up message (MaxRef portion defines matrix size)
                        PartMsgNum = 1; % checked
                        PartMsgLen = MsgLen - r; % checked

                        % from older part with unlimed (not 1000 part
message)

                        if r >= MaxPartLen

                            VehTX(idx+idx2+1:idx+idx2+MaxPartLen,x) =
PartMsg(1:MaxPartLen,1);  %
                            VehTX(idx+idx2+MaxPartLen+1:idx+idx2+r,x) =
MsgTemp(idx+idx2+MaxPartLen+1:idx+idx2+r,x);
                            PartMsg(1:MaxPartLen,1) =
MsgTemp(idx+idx2+1:idx+idx2+MaxPartLen,x);

                        else  % new portion added

                            VehTX(idx+idx2+1:idx+idx2+r,x) =
PartMsg(1:r,1);  %
                            MsgTemp(idx+idx2+1:idx+idx2+MaxPartLen-r,x)
= PartMsg(r+1:MaxPartLen,1);
                            PartMsg(1:MaxPartLen,1) =
MsgTemp(idx+idx2+1:idx+idx2+MaxPartLen,x);

                        end

                        ChanVehMatRem(CurrVeh,x) = 0;
                        idx = idx + r;

                    elseif MsgLen <= MaxRef %(MaxRef portion defines
matrix size)

                        VehTX(idx+idx2+1:idx+idx2+MaxPartLen,x)=
PartMsg(1:MaxPartLen,1);  % very new code
                        VehTX(idx+idx2+MaxPartLen+1:idx+idx2+MsgLen,x)=
MsgTemp(idx+idx2+MaxPartLen+1:idx+idx2+MsgLen,x);  % very new code
```

```
                        PartMsgNum = 0;
                        PartMsgLen = 0;
                        PartMsg = zeros(MaxPartLen,1);
                        ChanVehMatRem(CurrVeh,x) =
ChanVehMatRem(CurrVeh,x) - MsgLen;
                        idx = idx+MsgLen;

                    end;

                else

                    if(MsgLen > r && r <= MaxRef) % if only partial
room for partial, break up message (MaxRef portion defines matrix size)
                        PartMsgNum = 1; % checked
                        PartMsgLen = MsgLen - r; % checked
                        PartMsgNew(1:PartMsgLen,1) =
PartMsg(r+1:MsgLen,1); %PartMsgNew takes place of PartMsg; PartMsg
takes place of StatMsg
                        VehTX(idx+idx2+1:idx+idx2+r,x) =
PartMsg(1:r,1);  %

                        % restore PartMsg and PartMsgLen for use next
time
                        PartMsg = zeros(MaxPartLen,1);
                        PartMsg(1:PartMsgLen,1) =
PartMsgNew(1:PartMsgLen,1);
                        PartMsgNew = zeros(MaxPartLen,1);
                        ChanVehMatRem(CurrVeh,x) = 0;
                        idx = idx + r;

                    elseif MsgLen <= MaxRef %(MaxRef portion defines
matrix size)

                        VehTX(idx+idx2+1:idx+idx2+MsgLen,x)=
PartMsg(1:MsgLen,1);
                        PartMsgNum = 0;
                        PartMsgLen = 0;
                        PartMsg = zeros(MaxPartLen,1);
                        ChanVehMatRem(CurrVeh,x) =
ChanVehMatRem(CurrVeh,x) - MsgLen;
                        idx = idx+MsgLen;

                    end;
                end;

            elseif sum(MsgAttOut(:,3,2)) >=1%StatUMsgNum >=1


                StatUSeqNum = StatUSeqNum +1;
                %%%%%%%% below change important
                StatUSeqNumTx = mod(StatUSeqNum,256); % transmits 0-
255; starts at 1; vice 1-255
```

```
                SeqIdx = mod(StatUSeqNum-1,BuffSize)+1;  % verify +1;
starts at 1, range 1-1024

                MsgLen = MsgAttOut(SeqIdx,1,2);
                % Place MsgLen up here then replace MsgLenBi
                MsgVehBi = de2bi(MsgAttOut(SeqIdx,4,2),4,'left-msb')';
                MsgTypeBi = de2bi(2,4,'left-msb')';
                MsgLenBi = de2bi(MsgAttOut(SeqIdx,1,2)/8,16,'left-
msb')'; % and this

                StatUSeqNumBi = de2bi(StatUSeqNumTx,8,'left-msb')';   %
changed this too
                StatUTimeBi = de2bi(MsgAttOut(SeqIdx,2,2),16,'left-
msb')'; % and this

                ContUTypeBi = de2bi(3,8,'left-msb')';
                ContUNumBi = de2bi(ContUMsgNumTx,8,'left-msb')';
                ContUByteBi = de2bi(ContUByteNumTx,16,'left-msb')';
                RadTypeBi = de2bi(4,8,'left-msb')';
                RadNumBi = de2bi(RadMsgNumTx,8,'left-msb')';
                RadByteBi = de2bi(RadByteNumTx,16,'left-msb')';
                VidTypeBi = de2bi(5,8,'left-msb')';
                VidNumBi = de2bi(VidMsgNumTx,8,'left-msb')';
                VidByteBi = de2bi(VidByteNumTx,16,'left-msb')';

                MsgAttOut(SeqIdx,:,2) = [0 0 0 0];  % added this

                % StatUMsg should become StatUMsgHeader

StatUMsg=vertcat(MsgVehBi,MsgTypeBi,MsgLenBi,StatUSeqNumBi,StatUTimeBi,
ContUTypeBi,ContUNumBi,ContUByteBi,RadTypeBi,RadNumBi,RadByteBi,VidType
Bi,VidNumBi,VidByteBi);

                % header portion of MsgTemp should be zero'ed then
written to

                % Pad portion of MsgTemp should be taken from random
1's and 0's

                if MsgLen < MaxRef

                    MsgTemp(idx+idx2+1:idx+idx2+MsgLen,x) = StatUMsg;
% new code x

                end

                if(MsgLen > r && r <= MaxRef) % if only partial room,
break up message
                    PartMsgNum = 1;
                    PartMsgLen = MsgLen - r;
                    PartMsg(1:PartMsgLen,1) =
MsgTemp(idx+idx2+r+1:idx+idx2+MsgLen,x); % new code x
```

```
                    VehTX(idx+idx2+1:idx+idx2+r,x) =
MsgTemp(idx+idx2+1:idx+idx2+r,x); % new code x
                    ChanVehMatRem(CurrVeh,x) = 0;
                    idx = idx + r;

                elseif MsgLen <= MaxRef


VehTX(idx+idx2+1:idx+idx2+MsgLen,x)=MsgTemp(idx+idx2+1:idx+idx2+MsgLen,
x); % new code x
                    idx = idx+MsgLen;
                    ChanVehMatRem(CurrVeh,x) = ChanVehMatRem(CurrVeh,x)
- MsgLen;

                else

                end;


            elseif sum(MsgAttOut(:,3,3)) >=1%ContUMsgNum >=1

                ContUSeqNum = ContUSeqNum +1;
                %%%%%%%% below change important
                ContUSeqNumTx = mod(ContUSeqNum,256); % transmits 0-
255; starts at 1; vice 1-255
                SeqIdx = mod(ContUSeqNum-1,BuffSize)+1;  % verify +1;
starts at 1, range 1-1024

                MsgLen = MsgAttOut(SeqIdx,1,3);
                % Place MsgLen up here then replace MsgLenBi
                MsgVehBi = de2bi(MsgAttOut(SeqIdx,4,3),4,'left-msb')';
                MsgTypeBi = de2bi(3,4,'left-msb')';
                MsgLenBi = de2bi(MsgAttOut(SeqIdx,1,3)/8,16,'left-
msb')'; % and this
                ContUSeqNumBi = de2bi(ContUSeqNumTx,8,'left-msb')';   %
changed this too
                ContUTimeBi = de2bi(MsgAttOut(SeqIdx,2,3),16,'left-
msb')'; % and this
                MsgAttOut(SeqIdx,:,3) = [0 0 0 0];  % added this

                % ContUMsg should become ContUMsgHeader
                ContUMsgHdr =
vertcat(MsgVehBi,MsgTypeBi,MsgLenBi,ContUSeqNumBi,ContUTimeBi);

                % header portion of MsgTemp should be zero'ed then
written to

                % Pad portion of MsgTemp should be taken from random
1's and 0's

                MsgTemp(idx+idx2+1:idx+idx2+48,x) = ContUMsgHdr;  % new
code
```

```
                if(MsgLen > r && r <= MaxRef) % if only partial room,
break up message
                    PartMsgNum = 1;
                    PartMsgLen = MsgLen - r;
                    PartMsg(1:PartMsgLen,1) =
MsgTemp(idx+idx2+r+1:idx+idx2+MsgLen,x); % new code
                    VehTX(idx+idx2+1:idx+idx2+r,x) =
MsgTemp(idx+idx2+1:idx+idx2+r,x); % new code
                    ChanVehMatRem(CurrVeh,x) = 0;
                    idx = idx + r;

                elseif MsgLen <= MaxRef


VehTX(idx+idx2+1:idx+idx2+MsgLen,x)=MsgTemp(idx+idx2+1:idx+idx2+MsgLen,
x); % new code
                    idx = idx+MsgLen;
                    ChanVehMatRem(CurrVeh,x) = ChanVehMatRem(CurrVeh,x)
- MsgLen;

                else

                end;

            elseif sum(MsgAttOut(:,3,4)) >=1%RadMsgNum >=1

                RadSeqNum = RadSeqNum +1;
                %%%%%%%% below change important
                RadSeqNumTx = mod(RadSeqNum,256); % transmits 0-255;
starts at 1; vice 1-255
                SeqIdx = mod(RadSeqNum-1,BuffSize)+1;  % verify +1;
starts at 1, range 1-1024
                MsgLen = MsgAttOut(SeqIdx,1,4);
                % Place MsgLen up here then replace MsgLenBi
                MsgVehBi = de2bi(MsgAttOut(SeqIdx,4,4),4,'left-msb')';
                MsgTypeBi = de2bi(4,4,'left-msb')';
                MsgLenBi =
de2bi(floor(MsgAttOut(SeqIdx,1,4)/8),16,'left-msb')'; % and this
                RadSeqNumBi = de2bi(RadSeqNumTx,8,'left-msb')';   %
changed this too
                RadTimeBi = de2bi(MsgAttOut(SeqIdx,2,4),16,'left-
msb')'; % and this
                MsgAttOut(SeqIdx,:,4) = [0 0 0 0];  % added this
                % RadMsg should become RadMsgHeader
                RadMsgHdr =
vertcat(MsgVehBi,MsgTypeBi,MsgLenBi,RadSeqNumBi,RadTimeBi);

                % header portion of MsgTemp should be zero'ed then
written to
                % Pad portion of MsgTemp should be taken from random
1's and 0's

                MsgTemp(idx+idx2+1:idx+idx2+48,x) = RadMsgHdr;  % new
code x
```

```
                if(MsgLen > r && r <= MaxRef) % if only partial room,
break up message
                    PartMsgNum = 1;
                    PartMsgLen = MsgLen - r;

                    if PartMsgLen > MaxPartLen  % if greater than 1000
remainder

                        PartMsg(1:MaxPartLen,1) =
MsgTemp(idx+idx2+r+1:idx+idx2+r+MaxPartLen,x); % very new code
                        VehTX(idx+idx2+1:idx+idx2+r,x) =
MsgTemp(idx+idx2+1:idx+idx2+r,x); % very new code

                    else

                        PartMsg(1:PartMsgLen,1) =
MsgTemp(idx+idx2+r+1:idx+idx2+MsgLen,x); % new code x
                        VehTX(idx+idx2+1:idx+idx2+r,x) =
MsgTemp(idx+idx2+1:idx+idx2+r,x); % new code x

                    end

                    ChanVehMatRem(CurrVeh,x) = 0;
                    idx = idx + r;

                elseif MsgLen <= MaxRef


VehTX(idx+idx2+1:idx+idx2+MsgLen,x)=MsgTemp(idx+idx2+1:idx+idx2+MsgLen,
x); % new code
                    idx = idx+MsgLen;
                    ChanVehMatRem(CurrVeh,x) = ChanVehMatRem(CurrVeh,x)
- MsgLen;

                else

                end;

            elseif sum(MsgAttOut(:,3,5)) >=1%VidMsgNum >=1

                VidSeqNum = VidSeqNum +1;
                %%%%%%%% below change important
                VidSeqNumTx = mod(VidSeqNum,256); % transmits 0-255;
starts at 1; vice 1-255
                SeqIdx = mod(VidSeqNum-1,BuffSize)+1;  % verify +1;
starts at 1, range 1-1024
                MsgLen = MsgAttOut(SeqIdx,1,5);
                % Place MsgLen up here then replace MsgLenBi
                MsgVehBi = de2bi(MsgAttOut(SeqIdx,4,5),4,'left-msb')';
                MsgTypeBi = de2bi(5,4,'left-msb')';
                MsgLenBi =
de2bi(floor(MsgAttOut(SeqIdx,1,5)/8),16,'left-msb')'; % and this
```

```
                VidSeqNumBi = de2bi(VidSeqNumTx,8,'left-msb')';   %
changed this too
                VidTimeBi = de2bi(MsgAttOut(SeqIdx,2,5),16,'left-
msb')'; % and this
                MsgAttOut(SeqIdx,:,5) = [0 0 0 0];  % added this

                % VidMsg should become VidMsgHeader
                VidMsgHdr =
vertcat(MsgVehBi,MsgTypeBi,MsgLenBi,VidSeqNumBi,VidTimeBi);

                % header portion of MsgTemp should be zero'ed then
written to

                % Pad portion of MsgTemp should be taken from random
1's and 0's

                MsgTemp(idx+idx2+1:idx+idx2+48,x) = VidMsgHdr;  % new
code

                if(MsgLen > r && r <= MaxRef) % if only partial room,
break up message
                    PartMsgNum = 1;
                    PartMsgLen = MsgLen - r;

                    if PartMsgLen > MaxPartLen  % if greater than 1000
remainder

                        PartMsg(1:MaxPartLen,1) =
MsgTemp(idx+idx2+r+1:idx+idx2+r+MaxPartLen,x); % very new code
                        VehTX(idx+idx2+1:idx+idx2+r,x) =
MsgTemp(idx+idx2+1:idx+idx2+r,x); % very new code

                    else

                        PartMsg(1:PartMsgLen,1) =
MsgTemp(idx+idx2+r+1:idx+idx2+MsgLen,x); % new code x
                        VehTX(idx+idx2+1:idx+idx2+r,x) =
MsgTemp(idx+idx2+1:idx+idx2+r,x); % new code x

                    end

                    ChanVehMatRem(CurrVeh,x) = 0;
                    idx = idx + r;

                elseif MsgLen <= MaxRef



VehTX(idx+idx2+1:idx+idx2+MsgLen,x)=MsgTemp(idx+idx2+1:idx+idx2+MsgLen,
x); % new code x
                    idx = idx+MsgLen;
                    ChanVehMatRem(CurrVeh,x) = ChanVehMatRem(CurrVeh,x)
- MsgLen;
```

```
                else

                end

            else % if left over space and nothing to say then pad with
zeros

                % create pad message
                MsgLen = 208;  % change to ChanVehMatRem(CurrVeh,x)
with minimum of FillMsgHdr Length when expand msglen and also ensure
divisible by 8
                MsgVehBi = de2bi(CurrVeh,4,'left-msb')';
                MsgTypeBi = de2bi(9,4,'left-msb')';
                MsgLenBi = de2bi(MsgLen/8,16,'left-msb')';
                FillSeqNumBi = de2bi(FillSeqNum,8,'left-msb')';
                FillTimeBi = de2bi(CurrTime,16,'left-msb')';
                FillMsgHdr =
vertcat(MsgVehBi,MsgTypeBi,MsgLenBi,FillSeqNumBi,FillTimeBi);
                MsgTemp(idx+idx2+1:idx+idx2+48,x) = FillMsgHdr;  % new
code

                if(MsgLen > r) % if only partial room, break up message

                    PartMsgNum = 1;
                    PartMsgLen = MsgLen - r;
                    PartMsg(1:PartMsgLen,1) =
MsgTemp(idx+idx2+r+1:idx+idx2+MsgLen,x); % new code
                    VehTX(idx+idx2+1:idx+idx2+r,x) =
MsgTemp(idx+idx2+1:idx+idx2+r,x); % new code
                    ChanVehMatRem(CurrVeh,x) = 0;
                    idx = idx + r;

                else


VehTX(idx+idx2+1:idx+idx2+MsgLen,x)=MsgTemp(idx+idx2+1:idx+idx2+MsgLen,
x); % new code

                    idx = idx+MsgLen;
                    ChanVehMatRem(CurrVeh,x) = ChanVehMatRem(CurrVeh,x)
- MsgLen;

                end;
            end

        end
    end
end

HT = zeros(1,1);
EC = zeros(1,1);
Type = zeros(6,1);
```

```
LEN = zeros(16,1);
CID = zeros(16,1);
HCS = zeros(8,1);
HT = de2bi(0,1,'left-msb')';
EC = de2bi(1,1,'left-msb')';
Type = de2bi(4,6,'left-msb')'; % change for discard %%%%
CID = de2bi(CurrVeh+65280,16,'left-msb')';  %%%%%
HCS = de2bi(15,8,'left-msb')';  % not yet implememted

% append header if installed

for c=1:NumChan
    u=VehTXst(1,c);
    if(u>0)
        LEN = de2bi(((ChanVehMat(CurrVeh,c)+48)/8),16,'left-msb')';%
length in bytes including header, 2048 max, header is 6  %%%%
        VehTX(u:u+47,c)=vertcat(HT,EC,Type,LEN,CID,HCS);
    end
end

V1MsgCtOut(1,1) = 0;%StatUMsgNum;
V1MsgCtOut(2,1) = 0;%ContUMsgNum;
V1MsgCtOut(3,1) = 0;%RadMsgNum;
V1MsgCtOut(4,1) = 0;%VidMsgNum;  % 4 was ParMsgLen
V1MsgCtOut(5,1) = StatUSeqNum;
V1MsgCtOut(6,1) = ContUSeqNum;
V1MsgCtOut(7,1) = RadSeqNum;
V1MsgCtOut(8,1) = VidSeqNum;
V1MsgCtOut(9,1) = PartMsgLen;
V1MsgCtOut(10,1) = CurrTime;

V1MsgRemOut(1:MaxPartLen,1) = PartMsg(1:MaxPartLen,1);
```

4. SS MAC Transmitter

```
function [VehDataOut, outputFR, VehCh1, VehCh1Rate, VehCh1Snr, VehCh2,
VehCh2Rate, VehCh2Snr, VehCh3, VehCh3Rate,  VehCh3Snr, VehCh4,
VehCh4Rate, VehCh4Snr, VehCh5,VehCh5Rate,  VehCh5Snr, VehCh6,
VehCh6Rate, VehCh6Snr, VehCh7, VehCh7Rate, VehCh7Snr, VehCh8,
VehCh8Rate, VehCh8Snr]   = SSmacTX(ChanVehMatSnr, ChanRateIDU,
ChanVehMatBitUL, PacketDrop1, PacketDrop2, PacketDrop3, PacketDrop4,
PacketDrop5, PacketDrop6, Veh1TX, Veh2TX, Veh3TX, Veh4TX, Veh5TX,
Veh6TX)  % rate of transmission and transmission frame size
eml.extrinsic('de2bi');
% assume rateID to be static among chanel and for vehicle

NumChan = 4;
NumVeh = 6;

VehDataOut = zeros(NumVeh,26);

ULalloLen = 28512;
ULsymb = 6336;
```

```
DLalloLen = 3456;
DLsymb = 768;
MAPlen = 384;
MAPsymb = 768;

MessageTypes = 5;
packetdrop = zeros(NumVeh,MessageTypes);
packetdrop(1,:) = PacketDrop1;
packetdrop(2,:) = PacketDrop2;
packetdrop(3,:) = PacketDrop3;
packetdrop(4,:) = PacketDrop4;
packetdrop(5,:) = PacketDrop5;
packetdrop(6,:) = PacketDrop6;

outputFR= zeros(ULalloLen,NumChan);

% below is changed to boolean, might be a problem later

outputFR(:,1) =
Veh1TX(1:ULalloLen,1)|Veh2TX(1:ULalloLen,1)|Veh3TX(1:ULalloLen,1)|Veh4T
X(1:ULalloLen,1)|Veh5TX(1:ULalloLen,1)|Veh6TX(1:ULalloLen,1);
outputFR(:,2) =
Veh1TX(1:ULalloLen,2)|Veh2TX(1:ULalloLen,2)|Veh3TX(1:ULalloLen,2)|Veh4T
X(1:ULalloLen,2)|Veh5TX(1:ULalloLen,2)|Veh6TX(1:ULalloLen,2);
outputFR(:,3) =
Veh1TX(1:ULalloLen,3)|Veh2TX(1:ULalloLen,3)|Veh3TX(1:ULalloLen,3)|Veh4T
X(1:ULalloLen,3)|Veh5TX(1:ULalloLen,3)|Veh6TX(1:ULalloLen,3);
outputFR(:,4) =
Veh1TX(1:ULalloLen,4)|Veh2TX(1:ULalloLen,4)|Veh3TX(1:ULalloLen,4)|Veh4T
X(1:ULalloLen,4)|Veh5TX(1:ULalloLen,4)|Veh6TX(1:ULalloLen,4);

VehCh = zeros(ULalloLen,2*NumChan);
VehChRate = zeros(1,2*NumChan);
VehChSnr = zeros(1,2*NumChan);

x =0;
for c = 1:NumChan
    for v = 1:NumVeh
        if (ChanVehMatBitUL(v,c) > 0)
            x = x+1;
            VehCh(:,x) = outputFR(:,c)|outputFR(:,c);
            VehChSnr(1,x) = ChanVehMatSnr(v,c);
            VehChRate(1,x) = ChanRateIDU(1,c);

        end
    end
end

VehCh1 = VehCh(:,1)|VehCh(:,1);
VehCh2 = VehCh(:,2)|VehCh(:,2);
VehCh3 = VehCh(:,3)|VehCh(:,3);
VehCh4 = VehCh(:,4)|VehCh(:,4);
VehCh5 = VehCh(:,5)|VehCh(:,5);
VehCh6 = VehCh(:,6)|VehCh(:,6);
```

```
VehCh7 = VehCh(:,7)|VehCh(:,7);
VehCh8 = VehCh(:,8)|VehCh(:,8);

VehCh1Rate = VehChRate(1,1);
VehCh2Rate = VehChRate(1,2);
VehCh3Rate = VehChRate(1,3);
VehCh4Rate = VehChRate(1,4);
VehCh5Rate = VehChRate(1,5);
VehCh6Rate = VehChRate(1,6);
VehCh7Rate = VehChRate(1,7);
VehCh8Rate = VehChRate(1,8);

VehCh1Snr = VehChSnr(1,1);
VehCh2Snr = VehChSnr(1,2);
VehCh3Snr = VehChSnr(1,3);
VehCh4Snr = VehChSnr(1,4);
VehCh5Snr = VehChSnr(1,5);
VehCh6Snr = VehChSnr(1,6);
VehCh7Snr = VehChSnr(1,7);
VehCh8Snr = VehChSnr(1,8);

PkDropContDData = 16;
PkDropRadarData = 17;
PkDropVidData = 18;

for v = 1:NumVeh

    VehDataOut(v,PkDropContDData) = packetdrop(v,3);
    VehDataOut(v,PkDropRadarData) = packetdrop(v,4);
    VehDataOut(v,PkDropVidData) = packetdrop(v,5);

end
```

APPENDIX B — UV SENTRY FUNCTIONS

Listed in Table 23 are the modeled UV Sentry communications functions. A Core Workstation 6.0 file containing the functions listed below and also the functional relationships is available upon request from the Systems Engineering department.

Table 23. Modeled UV Sentry communications functions.

Number	Function
0.	Perform UV Sentry communications
1.	Perform BS application layer functions
1.1.	Transmit vehicle control message
1.1.1.	Transmit manual vehicle control message
1.1.2.	Transmit manual radar control message
1.1.3.	Transmit manual FLIR control message
1.1.4.	Transmit automatic vehicle control message
1.2.	Receive vehicle control message
1.2.1.	Receive equipment status message
1.2.2.	Receive equipment faults
1.2.3.	Receive fuel status
1.2.4.	Receive navigation information message
1.2.5.	Receive orientation information message
1.3.	Receive vehicle radar video message
1.4.	Receive vehicle FLIR video message
2.	Perform SS application layer functions
2.1.	Receive vehicle control message
2.1.1.	Receive manual vehicle control message
2.1.2.	Receive manual radar control message
2.1.3.	Receive manual FLIR control message
2.1.4.	Receive automatic vehicle control message
2.2.	Transmit vehicle control message
2.2.1.	Transmit equipment status message
2.2.2.	Transmit equipment faults
2.2.3.	Transmit fuel status
2.2.4.	Transmit navigation information message
2.2.5.	Transmit orientation information message
2.3.	Transmit vehicle radar video message
2.4.	Transmit vehicle FLIR video message

3.	Perform BS MAC functions
3.1.	Convert received bursts to application packets
3.1.1.	Receive UL transmission
3.1.2.	Read UL burst
3.1.2.1.	Locate UL burst in UL transmission
3.1.2.2.	Read UL burst header
3.1.2.2.1.	Read CID
3.1.2.2.2.	Read burst length
3.1.2.3.	Perform burst header error check
3.1.2.4.	Discard UL burst header
3.1.2.5.	Forward remaining UL burst to vehicle MAC assembly
3.1.3.	Assemble application packets
3.1.3.1.	Combine burst into superpacket
3.1.3.2.	Partition superpacket into application packets
3.1.3.3.	Read application packet header
3.1.3.3.1.	Read CID
3.1.3.3.2.	Read SFID
3.1.3.3.3.	Read length
3.1.3.3.4.	Read sequence number
3.1.3.3.5.	Read time
3.1.3.4.	Perform header check
3.1.3.5.	Discard packet header
3.1.4.	Forward application packet
3.1.4.1.	Perform latency check
3.1.4.1.1.	Check packet latency
3.1.4.1.2.	Discard late packet
3.1.4.2.	Forward to application packet to application
3.1.4.3.	Forward status packet to control
3.1.4.4.	(Relay) Forward to transmit
3.2.	Control MAC
3.2.1.	Receive application packet
3.2.2.	(Relay) Receive relay application packet
3.2.3.	Place application packet in SFID queue
3.2.4.	Perform queue management
3.2.5.	Record queued packet attributes
3.2.5.1.	Record original length
3.2.5.2.	Record original time
3.2.5.3.	Record original CID
3.2.5.4.	Record original SFID
3.3.	Allocate resources via MAP

3.3.1.	Allocate DL bursts
3.3.1.1.	Read CID/SFID queue sizes (total bytes)
3.3.1.2.	Determine CID priority
3.3.1.3.	Calculate CID/SFID total data requirements
3.3.1.4.	Determine CID rate ID
3.3.1.5.	Determine available BW
3.3.1.6.	(Relay) Determine relay cycle
3.3.1.7.	Assign DL bursts to CID
3.3.2.	Allocate UL bursts
3.3.2.1.	Read CID/SFID status packet
3.3.2.1.1.	Read SFID queue type
3.3.2.1.2.	Read queue length (number messages)
3.3.2.1.3.	Read SFID queue size (total bytes)
3.3.2.2.	Determine CID priority
3.3.2.3.	Calculate CID/SFID total data requirements
3.3.2.4.	Determine CID rate ID
3.3.2.5.	Determine available BW
3.3.2.6.	(Relay) Determine relay cycle
3.3.2.7.	Assign UL bursts to CID
3.3.3.	Create MAP
3.3.3.1.	Write DL burst assignments to MAP
3.3.3.1.1.	Write DL channel number
3.3.3.1.2.	Write DL vehicle rate ID
3.3.3.1.3.	Write DL CID
3.3.3.1.4.	Write DL CID allocation
3.3.3.2.	Write UL burst assignments MAP
3.3.3.2.1.	Write UL channel number
3.3.3.2.2.	Write UL vehicle rate ID
3.3.3.2.3.	Write UL CID
3.3.3.2.4.	Write UL CID allocation
3.3.4.	Send MAP to physical layer
3.4.	Convert packets to transmission bursts
3.4.1.	Create packet
3.4.1.1.	PoP priority application SFID packet
3.4.1.2.	Create packet header
3.4.1.2.1.	Write CID
3.4.1.2.2.	Write SFID
3.4.1.2.3.	Write length
3.4.1.2.4.	Write sequence number
3.4.1.2.5.	Write time

3.4.1.3.	Append header to packet
3.4.2.	Determine burst size
3.4.3.	Partition application packets
3.4.4.	Assemble burst
3.4.4.1.	Break last application packet into subpackets
3.4.4.2.	Assemble application packets and subpackets into burst
3.4.4.3.	Create UL burst header
3.4.4.3.1.	Write CID
3.4.4.3.2.	Write burst length
3.4.4.4.	Append burst header to burst
3.4.5.	Send burst to physical layer
4.	Perform SS MAC functions
4.1.	Convert received bursts to application packets
4.1.1.	Receive DL transmission from physical layer
4.1.2.	Read MAP
4.1.2.1.	Read DL vehicle rate ID
4.1.2.2.	Read DL CID
4.1.2.3.	Read DL CID allocation
4.1.2.4.	Read UL vehicle rate ID
4.1.2.5.	Read UL CID
4.1.2.6.	Read UL CID allocation
4.1.3.	Read DL burst
4.1.3.1.	Locate DL burst in DL transmission
4.1.3.2.	Read DL burst header
4.1.3.2.1.	Read CID
4.1.3.2.2.	Read burst length
4.1.3.3.	Perform burst header error check
4.1.3.4.	Discard DL burst header
4.1.3.5.	Forward remaining DL burst to vehicle MAC assembly
4.1.4.	Assemble application packets
4.1.4.1.	Combine bursts into superpacket
4.1.4.2.	Partition superpacket into application packets
4.1.4.3.	Read application packet header
4.1.4.3.1.	Read CID
4.1.4.3.2.	Read SFID
4.1.4.3.3.	Read length
4.1.4.3.4.	Read sequence number
4.1.4.3.5.	Read time
4.1.4.4.	Perform header check
4.1.4.5.	Discard packet header

4.1.5.	Forward application packet
4.1.5.1.	Forward to application
4.1.5.2.	(Relay) Forward to transmit
4.2.	Control MAC
4.2.1.	Receive application packet
4.2.2.	(Relay) Receive relay application packet
4.2.3.	Place application packet in SFID queue
4.2.4.	Perform queue management
4.2.4.1.	Check all queued packet latency
4.2.4.2.	Clear queue
4.2.5.	Record queued packet attributes
4.2.5.1.	Record original length
4.2.5.2.	Record original time
4.2.5.3.	Record original CID
4.2.5.4.	Record original SFID
4.3.	Convert packets to transmission bursts
4.3.1.	Create packet
4.3.1.1.	Pop priority application SFID packet
4.3.1.2.	Create network status packet
4.3.1.2.1.	Write SFID queue type
4.3.1.2.2.	Write SFID queue length (number messages)
4.3.1.2.3.	Write SFID queue size (total bytes)
4.3.1.3.	Create packet header
4.3.1.3.1.	Write CID
4.3.1.3.2.	Write SFID
4.3.1.3.3.	Write length
4.3.1.3.4.	Write sequence number
4.3.1.3.5.	Write time
4.3.1.4.	Append header to packet
4.3.2.	Determine burst size
4.3.3.	Assemble burst
4.3.3.1.	Break last application packet into subpackets
4.3.3.2.	Assemble application packets and subpackets into burst
4.3.3.3.	Create UL burst header
4.3.3.3.1.	Write CID
4.3.3.3.2.	Write burst length
4.3.3.4.	Append burst header to burst
4.3.4.	Send burst to physical layer
5.	Perform physical layer functions
5.1.	Convert data to symbols

5.1.1.	Insert convolution bits
5.1.2.	Punctured Reed-Solomon encode
5.1.3.	Convolution encode
5.1.4.	Interleave
5.1.5.	Modulate
5.1.6.	Insert nondata symbols
5.1.6.1.	Insert pilots symbols
5.1.6.2.	Insert guard symbols
5.1.6.3.	Insert preamble
5.2.	Inverse fast Fourier transform
5.3.	Transmit signal
5.4.	Simulate channel
5.4.1.	Adjust for Rician fading
5.4.2.	Adjust for Doppler spread
5.4.3.	Adjust for noise and interference
5.4.3.1.	Determine RX signal power
5.4.3.2.	Determine RX noise power
5.4.3.3.	Determine RX interference power
5.4.3.4.	Calculate SNIR
5.4.3.5.	Add noise and interference
5.5.	Receive signal
5.6.	Fast Fourier transform
5.7.	Convert symbols to data
5.7.1.	Remove nondata symbols
5.7.1.1.	Remove pilots symbols
5.7.1.2.	Remove guard symbols
5.7.1.3.	Remove preamble
5.7.2.	Demodulate
5.7.3.	Deinterleave
5.7.4.	Viterbi decode
5.7.5.	Punctured Reed-Solomon decode
5.7.5.	Remove convolution bits

LIST OF REFERENCES

[1] P. W. Singer, *Wired for War: The Robotics Revolution and Conflict in the Twenty-First Century,* 1st ed. New York: Penguin Press, 2009.

[2] Office of Naval Research, "UV Sentry system innovative naval prototype proposal overview," Office of Naval Research, Arlington, VA, Dec. 2008.

[3] M. Sherman, K. M. McNeill, K. Conner, P. Khuu and T. McNevin, "A PMP-Friendly MANET Networking Approach for WiMAX/IEEE 802.16/sup TM/," *IEEE Military Communications Conference*, pp. 1–7, 23–25, 2006.

[4] M.-T. Zhou, C.-W. Ang, P.-Y. Kong, J. Shankar, L. Zhang, R. Miura and M. Fujise, "Evaluation of the IEEE 802.16 Mesh MAC for Multihop Inter-ship Communications," *7th International Conference on ITS Telecommunications*, pp. 1–8, 2007.

[5] M. B. K. Hartzog and T. X. Brown, "Wimax - Potential Commercial Off-The-Shelf Solution for Tactical Mobile Mesh Communications," *IEEE Military Communications Conference*, pp. 1–7, 23–25, 2006.

[6] P. Sendín-Raña, F. J. González-Castaño, F. Gil-Castiñeira and P. S. Rodríguez-Hernández, "Dynamic multicast groups with adaptive path selection in MMR WiMAX networks," *IEEE Military Communications Conference*, pp. 1472–1477, 2010.

[7] D. H. Lee, S. C. Kim, D. C. Park, S. S. Hwang and Y. Kim, "Performance evaluation of multi-hop relay system with deployment scenarios," *IEEE Military Communications Conference*, pp. 1–7, 16–19, 2008.

[8] R. Amin, K. Wang and P. Ramanathan, "An Integrated Routing and Scheduling Approach for Persistent Vehicle Communication in Mobile WiMAX Mesh Networks," *IEEE Military Communications Conference*, pp. 1–7, 29–31, 2007.

[9] R. Olsen, C. Meagher, R. Ferro, C. de Jesus, A. Shum and S. Lapic, "Spatially Aware Wireless Networks (SPAWN) for Higher Data Rate and Range Performance with Lower Probability of Detection," *IEEE Military Communications Conference*, pp. 1–5, 29–31, 2007.

[10] J. G. Andrews, G. Arunabha and R. Muhamed, *Fundamentals of Wimax: Understanding Broadband Wireless Networking,* 1st ed. Upper Saddle River: Pearson Prentice Hall, 2007.

[11] T. R. Uecker, “Full motion video (FMV): The new dimension of imagery,” Master’s thesis, Air Command and Staff College, Maxwell Air Force Base, AL, 2005 [Online]. Available: https://www.afresearch.org/skins/rims/q_mod_be0e99f3-fc56-4ccb-8dfe-670c0822a153/q_act_downloadpaper/q_obj_cf02d5f1-0890-4ed5-aecc-133232745018/display.aspx?rs=enginespage, accessed Apr. 28, 2011.

[12] National Commission on the BP Deepwater Horizon Oil Spill and Offshore Drilling, “Deep water: The gulf oil disaster and the future of offshore drilling,” National Commission on the BP Deepwater Horizon Oil Spill and Offshore Drilling, Jan. 2001 [Online]. Available: http://www.oilspillcommission.gov/sites/default/files/documents/DEEPWATER_ReporttothePresident_FINAL.pdf, accessed Apr. 28, 2011.

[13] Australian Government Department of Defense, “Middle east task force command team returns home,” Australian Government Department of Defense, Jan. 2008 [Online]. Available: http://www.defence.gov.au/opex/global/opcatalyst/images/gallery/2008/0131/index.htm, accessed Apr. 28, 2011.

[14] Office of Naval Research, “UV Sentry design reference mission,” Office of Naval Research, Arlington, VA, Mar. 2009.

[15] Office of Naval Research, “UV Sentry: Illustrative operational vignettes,” Office of Naval Research, Arlington, VA, May 2009.

[16] Office of the Secretary of Defense, “Office of the Secretary of Defense unmanned systems roadmap,” Office of the Secretary of Defense, Washington D.C., Mar. 2009.

[17] Northrop Grumman, “MQ-8B fire scout,” Northrop Grumman [Online]. Available: http://www.as.northropgrumman.com/products/mq8bfirescout_navy/assets/firescout-new-brochure.pdf, accessed Apr. 28, 2011.

[18] Office of Naval Research, “Multi-target track and terminate (MT3),” Office of Naval Research, Arlington, VA, 2009 [Online]. Available: http://www.onr.navy.mil/~/media/Files/Funding-Announcements/BAA/09-023.ashx.

[19] Jane's, "Boghammar craft," Jane's, Dec. 2010 [Online]. Available: http://search.janes.com.libproxy.nps.edu/Search/documentView.do?docId=/content1/janesdata/yb/jfs/jfs_1538.htm@current&pageSelected=allJanes&keyword=boghammarspeed&backPath=http://search.janes.com.libproxy.nps.edu/Search&Prod_Name=JFS&, accessed Apr. 28, 2011.

[20] British Broadcasting Corporation, "Nigerian attack closes oilfield," *BBC News*, June 2008 [Online]. Available: http://news.bbc.co.uk/2/hi/africa/7463288.stm, accessed Apr. 28, 2011.

[21] British Broadcasting Corporation, "Speedboat attack on Nigeria rig," *BBC News,* Jan. 2006 [Online]. Available: http://news.bbc.co.uk/2/hi/africa/4615272.stm, accessed Apr. 28, 2011.

[22] D. A. Moseley, "Motion Imagery Interpretability: the Digital Age of Intelligence, Surveillance, and Reconnaissance," Master's thesis, University of Denver University College, Denver, CO, 2010 [Online]. Available: http://ectd.du.edu/source/uploads/21653235.pdf, accessed Apr. 28, 2011.

[23] L. P. McCaskill, "Unmanned vehicle sentry architecture: Interim delivery," WBB Consulting, Reston, VA, Mar. 2010.

[24] Unites States Navy, "Universal naval task list (UNTL)," Jan. 2007 [Online]. Available: http://doni.dla.mil, accessed Apr. 28, 2011.

[25] L. Kleinrock and F. Tobagi, "Packet Switching in Radio Channels: Part I--Carrier Sense Multiple-Access Modes and Their Throughput-Delay Characteristics," *Communications, IEEE Transactions on*, vol. 23, no. 12, pp. 1400–1416, Dec. 1975.

[26] S. W. Peters and R. W. Heath, "The future of WiMAX: Multihop relaying with IEEE 802.16j," *Communications Magazine, IEEE*, vol. 47, no. 1, pp. 104–111, Jan. 2009.

[27] I. F. Akyildiz and X. Wang, "A survey on wireless mesh networks," *Communications Magazine, IEEE*, vol. 43, no. 9, pp. S23–S30, Sept. 2005.

[28] D. Agrawal and Q. Zeng, *Introduction to Wireless and Mobile Systems*, 3rd ed. Stamford: Cenegate Learning, 2010.

[29] V. Rajendran, K. Obraczka and J. J. Garcia-Luna-Aceves, "Energy-efficient, collision-free medium access control for wireless sensor networks," *Wirel. Netw.*, vol. 12, no. 1, pp. 63–78, Feb. 2006.

[30] N. Abramson, “The ALOHA System – Another Alternative for Computer Communications,” *Proc. of the AFIPS Fall Joint Computer Conf.*, vol. 37, pp. 281–285, 1970.

[31] P. Karn, “MACA—A new channel access method for packet radio,” *Proc. 9th ARRL Comput. Netw. Conf.*, pp. 134–140, 1990.

[32] “IEEE Standard for Information technology – Telecommunications and information exchange between systems – Local and metropolitan area networks – Specific requirements – Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications,” *IEEE Std 802.11-2007 (Revision of IEEE Std 802.11-1999)*, Jun. 12, 2007.

[33] S. Heier, C. Ellerbrock and M. Malkowski, "UMTS medium access control quality of service scheduling," *The 13th IEEE International Symposium on Personal, Indoor and Mobile Radio Communications*, vol. 4, pp. 15–18, 1893–1897, 2002.

[34] "IEEE Standard for Local and metropolitan area networks Part 16: Air Interface for Broadband Wireless Access Systems," *IEEE Std 802.16-2009 (Revision of IEEE Std 802.16-2004*, pp. C1–2004, May 29 2009.

[35] "IEEE Standard for Local and metropolitan area networks Part 16: Air Interface for Broadband Wireless Access Systems Amendment 1: Multiple Relay Specification," *IEEE Std 802.16j-2009 (Amendment to IEEE Std 802.16-2009)*, pp. C1–290, June 12, 2009.

[36] Harris Communications, “RT-1944/U SeaLancet™,” Harris Communications, 2010 [Online]. Available: http://download.harris.com/app/public_download.asp?fid=2063, accessed Apr. 28, 2011.

[37] Harris Communications, “SeaLancet™ RT-1944/U IP network radio,” Harris Communications, Jan. 2009 [Online]. Available: http://download.harris.com/app/public_download.asp?fid=2101, accessed Apr. 28, 2011.

[38] Alvarion, “OFDM overview: White paper,” Alvarion, Sept. 2003 [Online]. Available: http://www.sparcotech.com/Alvarion%20-%20Orthagonal%20Freq%20Div%20Multiplexing.pdf, accessed Apr. 28, 2011.

[39] CelPlan Technologies, “Designing WiMAX networks,” CelPlan Technologies, Mar. 2008 [Online]. Available: http://www.celplan.com/WhitePaper/WhitePaperRegister.asp, accessed Apr. 28, 2011.

[40] WiMAX Forum, “Mobile WiMAX – part I: A technical overview and performance evaluation.” WiMAX Forum, 2006 [Online]. Available: http://www.wimaxforum.org/technology/downloads/Mobile_WiMAX_Part1_Overview_and_Performance.pdf, accessed Apr. 28, 2011.

[41] OPNET, "WiMAX Network Simulation with OPNET," OPNET, 2010 [Online]. Available: http://www.opnet.com/solutions/brochures/WiMAX_Network_Simulation_with_OPNET_Modeler.pdf, accessed Apr. 28, 2011.

[42] QualNet, “Advanced wireless library (WiMAX*),” QualNet, Apr. 2010 [Online]. Available: http://www.scalable-networks.com/wp-content/uploads/2010/05/SNT_wiMAX_web.pdf, accessed Apr. 28, 2011.

[43] The MathWorks, "IEEE® 802.16-2004 OFDM PHY Link, Including Space-Time Block Coding," MATLAB R2010a, 2009.

[44] Z. Fengqing and K. Quancun, "Distributed Remote Control System of UAV Based on Man-in-loop Real-time Operation," *Chinese Control Conference*, pp. 119–122, 2007.

[45] FLIR Systems, “Star SAFIRE III datasheet,” FLIR Systems [Online]. Available: http://www.gs.flir.com/uploadedFiles/GS/datasheets/Brochure_StarSAFIREIII.pdf, accessed Apr. 28, 2011.

[46] Y. M. Chen, L. Dong and J.-S. Oh, "Real-Time Video Relay for UAV Traffic Surveillance Systems Through Available Communication Networks," *IEEE Wireless Communications and Networking Conference*, pp. 2608–2612, 2007.

[47] Furuno, “Model 1832,” Furuno [Online]. Available: http://www.furunousa.com/ProductDocuments/1832%20Brochure.pdf, accessed Apr. 28, 2011.

[48] L. E. Russo, "Space-time compression of FLIR video," *International Conference on Image Processing, Proceedings*, vol. 2, pp. 419–422, 2000.

[49] V. Sabbatino and S. Bottalico, "Radar image compression," *Radar 97 (Conf. Publ. No. 449)*, pp. 725–728, 1997.

[50] A. Chodorek and R. R. Chodorek. "An MPEG-2 video traffic prediction based on phase space analysis and its application to on-line dynamic bandwidth allocation," *2nd European Conference on Universal Multiservice Networks*, pp. 44–55, 2002.

[51] T. Le-Ngoc and S.N. Subramanian, "A Pareto-modulated Poisson process (PMPP) model for long-range dependent traffic," *Computer Communications,* vol. 23, pp. 123–132, 15 Jan. 2000.

[52] J. Joe, S. K. Hazra, S. H. Toh, W. M. Tan and J. Shankar, "5.8 GHz Fixed WiMAX Performance in a Sea Port Environment," *IEEE 66th Vehicular Technology Conference*, pp. 879–883, 2007.

[53] K. Yang, T. Røste, F. Bekkadal and T. Ekman, "Land-to-Ship Radio Channel Measurements over Sea at 2 GHz," *6th International Conference on Wireless Communications Networking and Mobile Computing (WiCOM)*, pp. 1–4, 23–25, 2010.

[54] K. Maliatsos, P. Loulis, M. Chronopoulos, P. Constantinou, P. Dallas and M. Ikonomou, "Experimental Small Scale Fading Results for Mobile Channels Over the Sea," *IEEE 17th International Symposium on Personal, Indoor and Mobile Radio Communications*, pp. 1–5, 11–14, 2006.

[55] M. W. Baidas and T. O'Farrell, "The Performance of Coded NonCoherent M-ary Orthogonal Keying Based OFDM Systems in a Frequency Selective and Fast Time-Varying Channel," *IEEE International Conference on Communications*, pp. 2930–2935, 2007.

[56] W. C. Jakes, *Microwave Mobile Communications,* West Sussex: John Wiley & Sons, 1975.

[57] A. E. Leu, B. L. Mark and M. A. McHenry, "A Framework for Cognitive WiMAX With Frequency Agility," *Proceedings of the IEEE*, vol. 97, no. 4, pp. 755–773, 2009.

INITIAL DISTRIBUTION LIST

1. Defense Technical Information Center
   Ft. Belvoir, Virginia

2. Dudley Knox Library
   Naval Postgraduate School
   Monterey, California

3. LT Nathan Matson, USN
   Naval Postgraduate School
   Monterey, California

4. Dr. Clifford Whitcomb
   Naval Postgraduate School
   Monterey, California

5. Dr. Weilian Su
   Naval Postgraduate School
   Monterey, California

6. Dr. R. Clark Robertson
   Naval Postgraduate School
   Monterey, California